
INPUT
Publicación práctica
para usuarios de
sinclair
Revista mensual 1987
Precio 375 Ptas
Año 2 Número 20
MAPA Y CARGADOR
DEL
FUTURE KNIGHT
UDGS: CREA TU ZOO FANTASTICO
RUTINAS DE CARGA DEL SISTEMA
Y ADEMAS, NUESTRA SECCION DE JUEGOS



AÑO 2 NUMERO 20

DIRECTOR: Manuel Pérez
DIRECTOR DE ARTE: Luis F. Balaguer
REALIZACION GRAFICA: Didac Tudela
COLABORADORES: José Vila, Antonio Pliego, Xavier Ferrer, Ernesto del Valle, Equipo Molisoft, Ramón Rabaso, Antonio Taratiel, Jaime Mardones, Carles Bartolomé, Angels Alvarez
FOTOGRAFIA: Ernesto Wallisch, Joan Boada

INPUT Sinclair es una publicación de PLANETA-DE AGOSTINI, S.A.
GERENTE DIVISION DE REVISTAS:
Sebastián Martínez

PUBLICIDAD: José Real-Grupo Jota
Madrid: c/ General Varela, 35
Teléf. 270 47 02/03
Barcelona: Avda. de Sarriá, 11-13, 1.°
Teléf. 250 23 99

FOTOMECANICA: TECFA, S.A.

IMPRESION: Sirven Grafic
c/ Gran Vía, 754-756, 08013 Barcelona
Depósito legal: B. 38.115-1986

SUSCRIPCIONES: EDISA
López de Hoyos, 141, 28002 Madrid
Teléf. (91) 415 97 12

REDACCION:
Aribau, 185, 1.°
08021 Barcelona

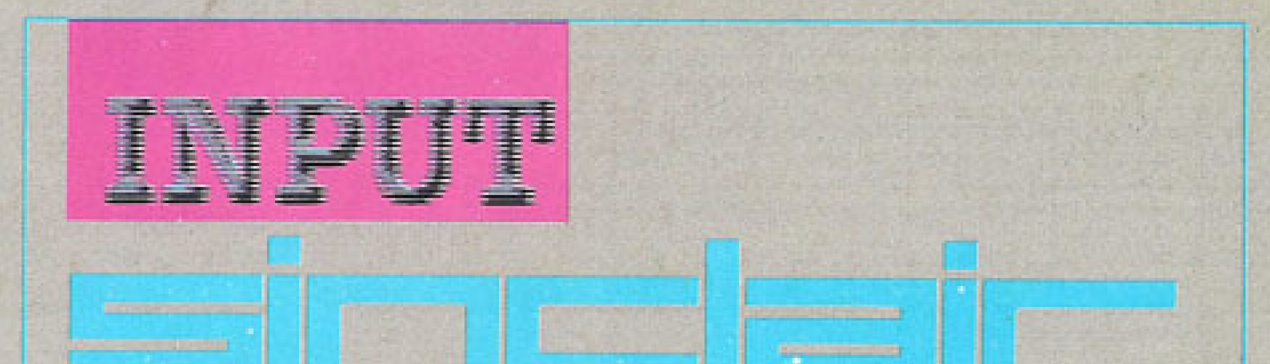
DISTRIBUIDORA:
R.B.A. PROMOTORA DE EDICIONES, S.A.
Calle B, n.° 11, Sector B, Zona Franca
08004 Barcelona

El precio será el mismo para Canarias que para la Península y en él irá incluida la sobretasa aérea.

INPUT Sinclair es una publicación controlada por

INPUT Sinclair es independiente y no está vinculada a Sinclair Research o sus distribuidores.

INPUT no mantiene correspondencia con sus lectores, si bien la recibe, no responsabilizándose de su pérdida o extravío. Las respuestas se canalizarán a través de las secciones adecuadas en estas páginas.



SUMARIO

EDITORIAL

BUENOS MOMENTOS

Un cualificado representante de una de las principales empresas españolas distribuidoras de videojuegos nos comunicaba recientemente que el movimiento en esta franja del software para los microordenadores domésticos estaba superando todas las previsiones.

Evidentemente estas previsiones estaban referidas al reciente descenso de sus precios de venta al público.

Como ya habíamos comentado en estas mismas líneas en otras ocasiones, tal medida no debía limitarse a una simple operación de marketing destinada a invertir la relación entre el número de ejemplares vendidos y su precio unitario.

Así, el objetivo final debía ser el de definir una terapia de conjunto para el sector. Alcanzar un porcentaje de producción propia significativo, responder a las características específicas del mercado nativo, y fomentar el surgimiento de nutridos y sólidos grupos de programadores son un objetivo obligado de cualquiera que pretenda que el software en nuestro país deje de ser un vergonzoso, y exclusivo, problema de aduanas, licencias e impuestos.

En este cuadro, el papel que corresponde jugar a las revistas del sector debería modificarse también. Existe la necesidad de que los lectores de esas revistas y usuarios por tanto de los diferentes sistemas, identifiquen a los medios de comunicación como analistas veraces, críticos severos y cualificadores objetivos. Sólo cumpliendo estos requisitos serán portavoces reputados por sus valoraciones. Frente a los usuarios por su fiabilidad a la hora de inclinarse por una u otra opción. Frente a los productores y distribuidores por su análisis ajeno a la tendenciosidad o a una mal comprendida indulgencia.

Dicho esto, también debiera decirse que la problemática del software para nuestros sistemas no debe ni puede reducirse al subsector de los videojuegos. En un momento en que la informática invade más y más áreas de la actividad social y, entre ellas, la educativa, no debe pensarse sólo en fomentar una utilización unilateral de los pequeños, entre los más pequeños, micros.

Que productores y distribuidores se plantearan un apoyo consciente, técnico, económico y publicitario, a los diferentes tipos de software educativo y de aplicaciones sería también una manera de elevar el nivel de los usuarios y su capacidad de valorar críticamente sus propias actividades. Además de necesario sería muy sano.

LAS RUTINAS DE CARGA DEL SISTEMA OPERATIVO

- LA RUTINA PRINCIPAL DE CARGA
- NOTAS ACLARATORIAS
- MAYOR PROTECCION DE TUS PROGRAMAS

Para completar nuestro informe acerca de las rutinas de carga y grabación del sistema operativo, te ofrecemos hoy esta segunda parte que esperamos te sirva como adecuado complemento a este pequeño estudio.

Comenzaremos con una breve exposición de la estructura y disposición de las rutinas en ROM, para continuar con unos apuntes acerca del tema de su manipulación (cambio de los colores de carga, utilización de cabeceras falsas, carga de bloques grabados con distinto tono...).

LA RUTINA PRINCIPAL DE CARGA

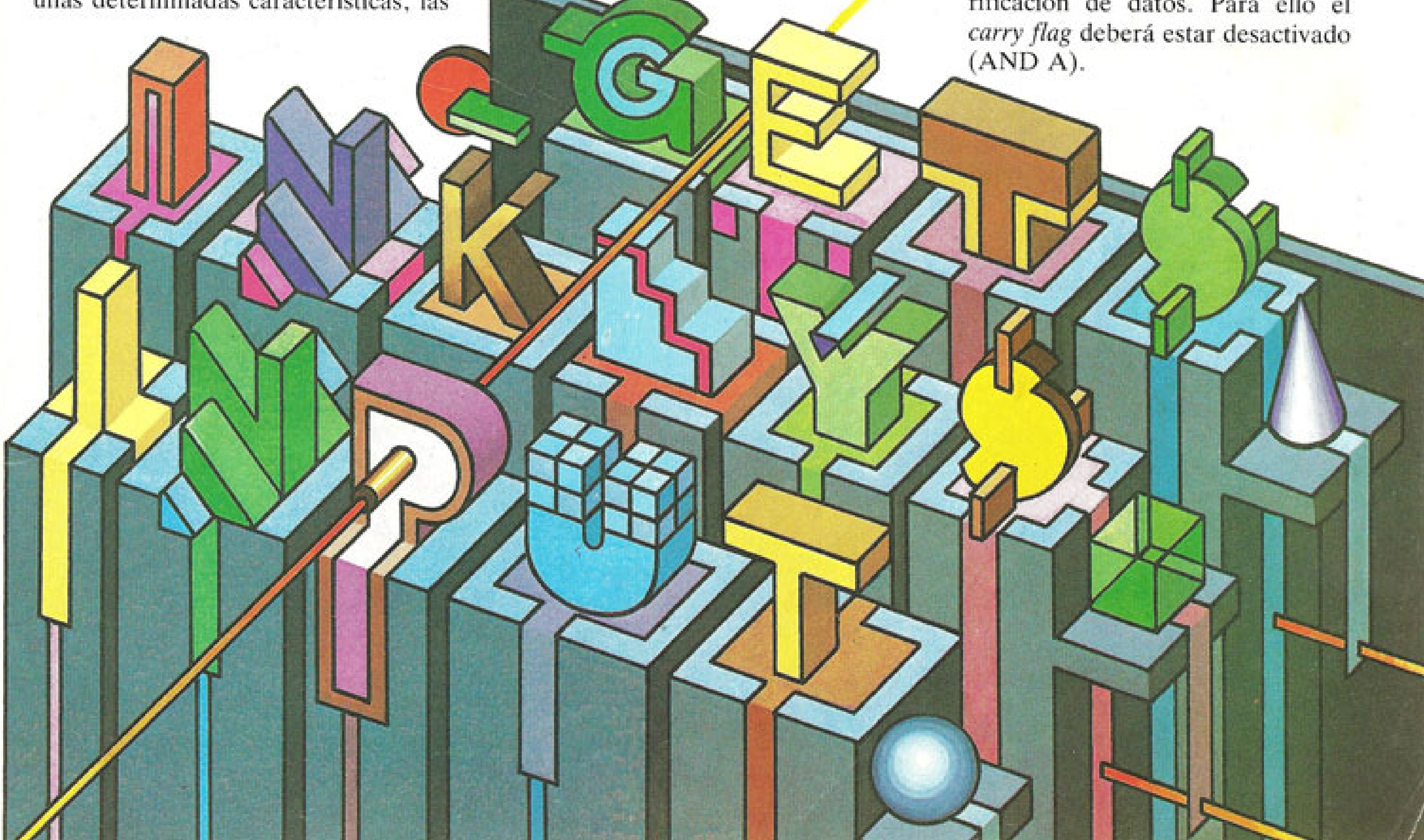
Sin lugar a dudas la rutina más sencilla de utilizar desde el código máquina es la ubicada en la dirección de memoria #0556 (1366 en decimal). Su uso nos permite cargar de una manera rápida y eficaz un bloque de bytes de unas determinadas características, las cuales seleccionaremos introduciendo una serie de valores en los registros pertinentes.

A grandes rasgos, podemos decir que la rutina se extiende desde la mencionada posición hexadecimal #0556 a la #05E2 inclusive, y, por tanto, ocupa 141 bytes de longitud. Cuando utilizamos el comando BASIC LOAD en cualquiera de sus formas, la rutina que realmente utilizamos es la situada en la posición #0808, que es la que gestiona el proceso de carga, pero en el fondo lo que hace es organizar los datos de la cabecera para posteriormente llamar a la #0556. Esto en principio puede parecer un poco contradictorio, pero estudiaremos la #0556 porque además de darnos mayor libertad (para cargar un bloque no debemos ceñirnos a los datos de la cabecera), es mucho más sencilla de usar.

Los parámetros que necesita la rutina para su correcto funcionamiento se han de entrar en estos registros:

— Cargaremos el registro A con un número acorde con el tono del bloque que deseamos cargar. El número 0 es el tono elegido por el sistema operativo para distinguir las cabeceras. Si, por el contrario, cargamos el acumulador con el número #FF (255), normalmente el sistema operativo entenderá que se trata de un bloque de bytes que nada tiene que ver con la cabecera, sino más bien con el cuerpo principal del programa.

— Si antes de iniciar el proceso de carga activamos el *carry flag* (SCF), la rutina #0556 entenderá que lo que se desea es iniciar un proceso de carga. Otra posibilidad que se nos ofrece es la de simplemente activar la rutina para la verificación de datos. Para ello el *carry flag* deberá estar desactivado (AND A).



— El registro indicador IX deberá almacenar la posición de la memoria que queramos que ocupen los datos a cargar desde el cassette.
— El registro doble DE será el indicador de la longitud de los datos a cargar.
— La llamada a la rutina se ejecutará con el acostumbrado CALL.

Como una serie de notas aclaratorias podemos decir:
— Si estamos llevando a cabo una verificación, el proceso se detendrá si los datos no coinciden, y, aunque muestre el mensaje OK, notaremos que el cassette sigue volcando datos («sigue el ruido»), como en el caso «R Tape Loading Error» de la verificación desde el BASIC.
— Si intentamos cargar más datos de los indicados en el registro DE, el proceso terminará sin que se produzca error alguno al llegar al número especificado, aun cuando queden una cierta cantidad sin leer de la cinta.
— Si, por el contrario, el bloque a cargar del cassette es de longitud inferior a la indicada por DE, el proceso acabará también sin error cuando se lea el último dato de la cinta (como comprobarás, esta situación y las dos anteriores no se cumplen cuando trabajamos desde el BASIC, generando el correspondiente mensaje de error).
— Deberemos tener un cierto conocimiento previo acerca de lo que se desea cargar. Si queremos cargar un bloque de bytes con tono #FF pero en la cinta se encuentra antes un bloque distinto, como, por ejemplo, una cabecera (17 bytes grabados con tono 0), el proceso se detendrá y no habremos conseguido nada. El cassette debe estar posicionado en el comienzo justo del bloque que queremos leer.

Todas estas puntualizaciones que en un principio parecen un poco restrictivas, veremos cómo luego se vuelven a nuestro favor (podremos manejar bloques de datos sin importarnos las cabeceras, o incluso falsear éstas para una mejor protección de nuestros programas).

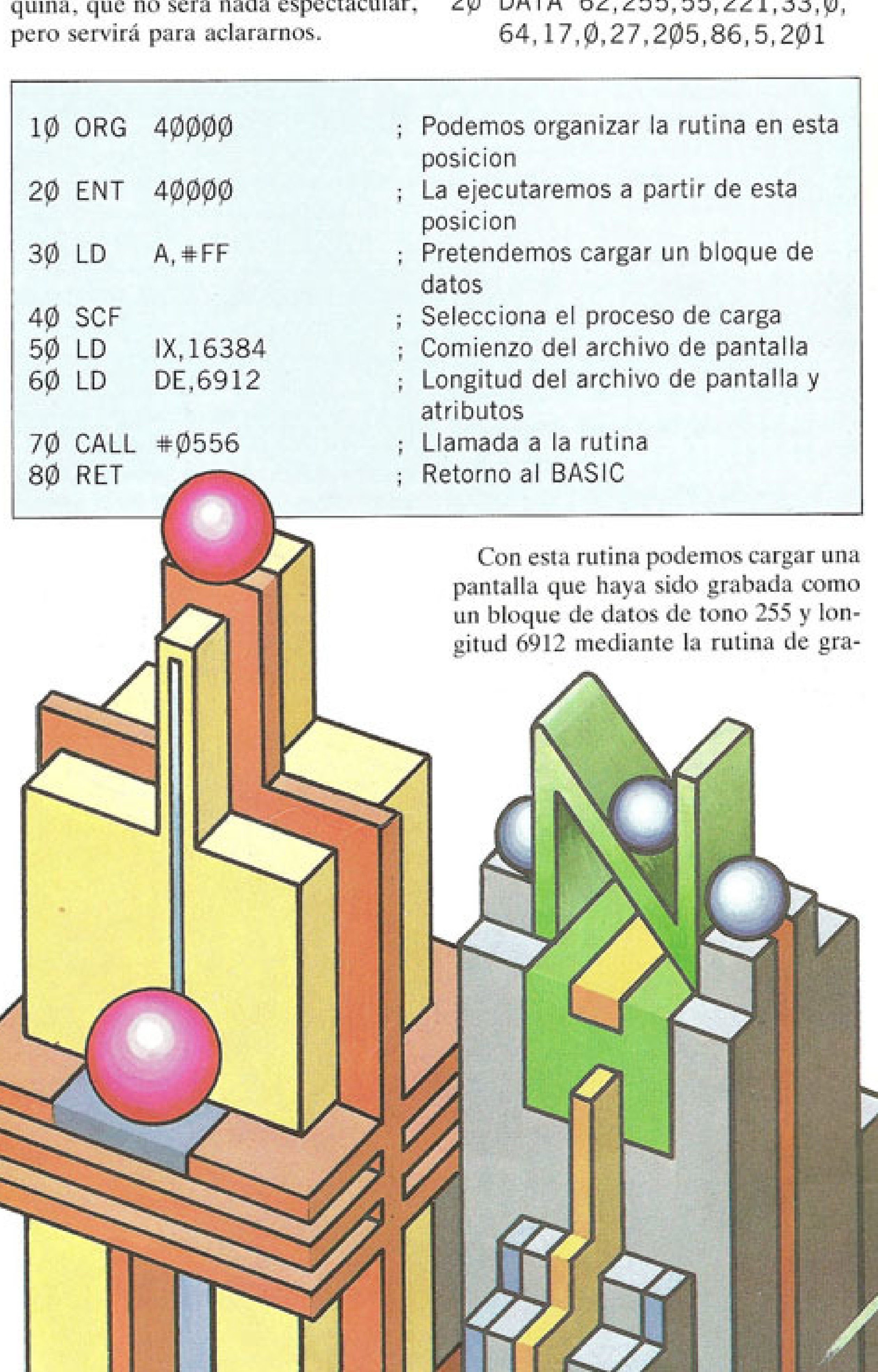
Centrándonos en la utilización de la rutina, y para comprobar lo arriba expuesto, podemos usar una rutina de carga de pantallas desde el código máquina, que no será nada espectacular, pero servirá para aclararnos.

El cargador BASIC de la rutina quedará como:

```
1Ø CLEAR 39999: FOR
   N=4ØØØØ TO 4ØØ13: READ
   A: POKE N,A: NEXT N
2Ø DATA 62,255,55,221,33,Ø,
   64,17,Ø,27,2Ø5,86,5,2Ø1
```

```
1Ø ORG  4ØØØØ      ; Podemos organizar la rutina en esta
                     posicion
2Ø ENT  4ØØØØ      ; La ejecutaremos a partir de esta
                     posicion
3Ø LD   A,#FF      ; Pretendemos cargar un bloque de
                     datos
4Ø SCF             ; Selecciona el proceso de carga
5Ø LD   IX,16384   ; Comienzo del archivo de pantalla
6Ø LD   DE,6912    ; Longitud del archivo de pantalla y
                     atributos
7Ø CALL #Ø556      ; Llamada a la rutina
8Ø RET             ; Retorno al BASIC
```

Con esta rutina podemos cargar una pantalla que haya sido grabada como un bloque de datos de tono 255 y longitud 6912 mediante la rutina de gra-



bación #04C2. También podremos cargar cualquier pantalla que hayamos grabado desde el BASIC con SAVE «nombre» SCREEN$, siempre que tengamos en cuenta las excepciones antes mencionadas. Con esto queremos decir que si rodamos el cargador BASIC y ejecutamos la rutina con RANDOMIZE USR 40000, al poner en marcha el cassette, lo que se espera encontrar es el bloque con los datos de la pantalla, pero muy a pesar nuestro antes viene la cabecera grabada con tono 0, lo cual detiene el proceso iniciado sin conseguir el resultado apetecido. Hay dos maneras de evitar que ocurra esto:

— Como antes hemos indicado, colocar la cinta con los datos justo después de la información de la cabecera.
— También podemos, por así decirlo, «desechar» la información de la cabecera repitiendo el proceso anterior dos veces, es decir, cargar primero un bloque de 17 bytes con tono 0 (cabecera), aun a sabiendas de que no lo vamos a utilizar, y a continuación cargar la información acerca de la pantalla. Con esta última modificación nuestra rutina queda:

```
 10 ORG  40000    ; Como el lector podrá comprobar, ya
 20 ENT  40000    ; que no vamos a utilizar el bloque de
 30 LD   A,#00    ; 17 bytes, como no queremos que
 40 SCF           ; puedan interferir en alguna de las
 50 LD   IX,0     ; posiciones de la RAM ocupadas, lo
 60 LD   DE,17    ; cargamos directamente en la
 70 CALL #0556    ; posición 0 de la ROM, lo cual no
 80 LD   A,#FF    ; causa ningún efecto extraño. El resto
 90 SCF           ; del proceso es igual al antes
100 LD   IX,16384   indicado, con lo cual simplemente
110 LD   DE,6912    hemos eludido ese posible defecto.
120 CALL #0556
130 RET
```

El cargador BASIC quedará como:

```
10 CLEAR 39999: FOR
   N=40000 TO 40026: READ
   A: POKE N,A: NEXT N
20 DATA 62,0,55,221,33,0,0,
   17,17,0,205,86,5
30 DATA 62,255,55,221,33,0,
   64,17,0,27,205,86,5,201
```

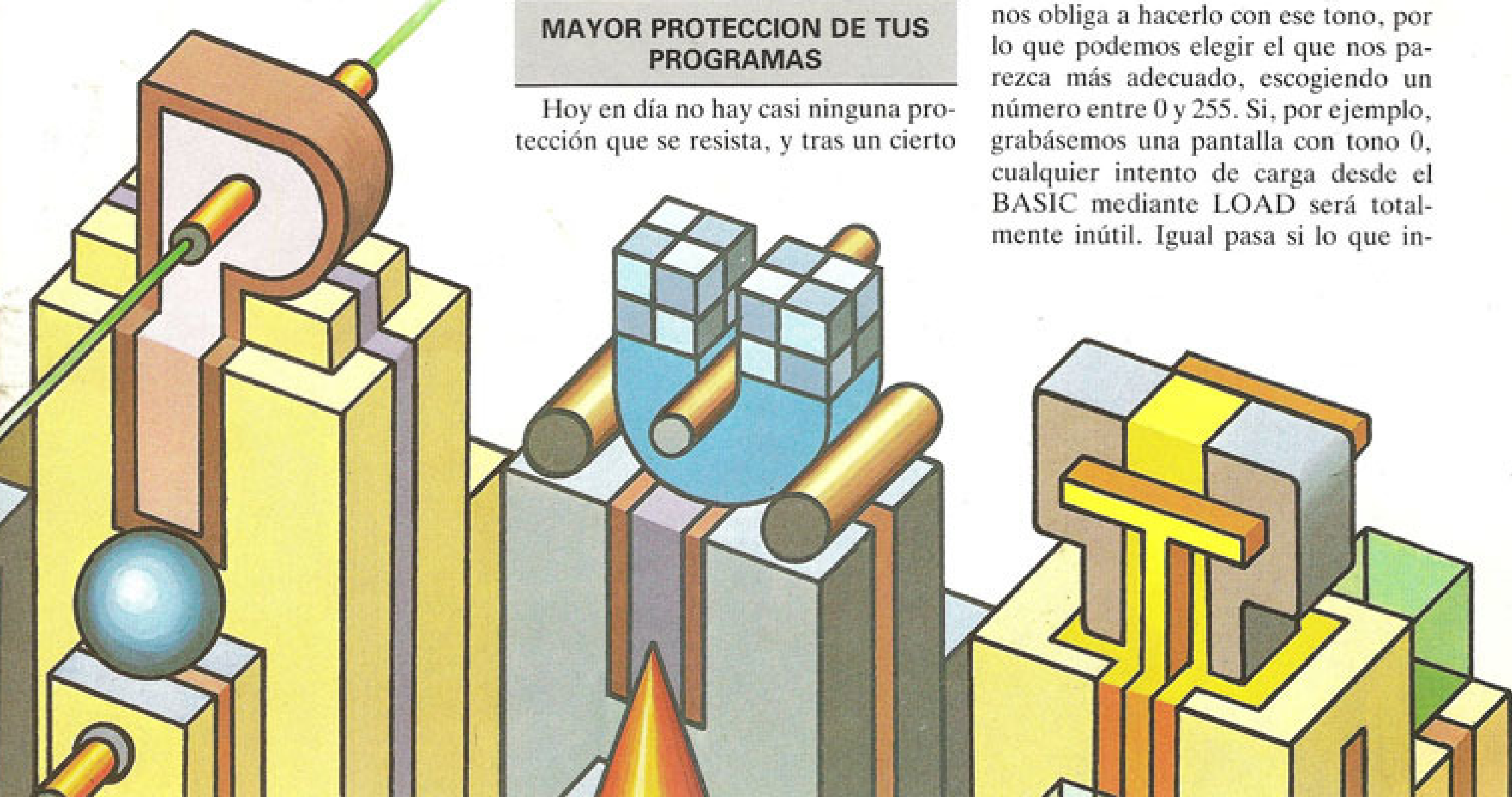
Ahora si después de rodar el cargador ejecutamos la rutina con RANDOMIZE USR 40000, comprobaremos que sí podemos cargar directamente una pantalla grabada desde el BASIC, aunque lo hayamos hecho de manera un poco distinta (pero imperceptible a los ojos de los demás).

MAYOR PROTECCION DE TUS PROGRAMAS

Hoy en día no hay casi ninguna protección que se resista, y tras un cierto tiempo, todas acaban saltando. Hay que recordar que nuestro Spectrum lleva algunos años en el mercado, y los programadores cada vez lo conocen mejor. Lo que aquí te proponemos no es nada espectacular, y seguramente no evitará todos tus problemas de protección, pero seguro que también evita las «miradillas» de muchos curiosos.

Como antes habíamos mencionado, aunque en un principio puede resultar más «engorroso» grabar tus programas desde c/m, también ofrecía ciertas ventajas.

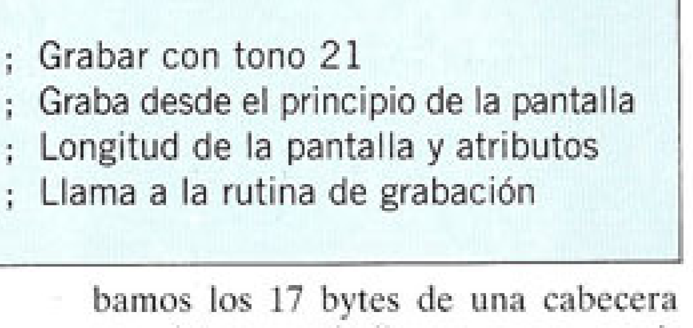
Hasta ahora por seguir un poco los pasos del sistema operativo, habíamos grabado los datos con tono 255, pero lo que también es cierto es que nadie nos obliga a hacerlo con ese tono, por lo que podemos elegir el que nos parezca más adecuado, escogiendo un número entre 0 y 255. Si, por ejemplo, grabásemos una pantalla con tono 0, cualquier intento de carga desde el BASIC mediante LOAD será totalmente inútil. Igual pasa si lo que in-



tentamos es cargar desde el BASIC una cabecera que ha sido grabada con tono 255. Con esto conseguimos que aparte de ser los únicos que conozcamos las verdaderas dimensiones de nuestros bloques de datos en código máquina, sólo podamos nosotros diseñar la rutina de carga adecuada para dicho bloque.

A modo de muestra, he aquí el ejemplo apuntado:

— Para grabar la pantalla con un tono 21 utilizaremos la rutina que ya explicamos anteriormente, la #04C2:



```
10  ORG  50000
20  ENT  50000
30  LD   A,#21        ; Grabar con tono 21
40  LD   IX,16384     ; Graba desde el principio de la pantalla
50  LD   DE,6912      ; Longitud de la pantalla y atributos
60  CALL #04C2        ; Llama a la rutina de grabación
70  RET
```

El cargador BASIC quedará:

```
10 CLEAR 49999: FOR
   N=50000 TO 50012: READ
   A: POKE N,A: NEXT N
20 DATA 62,21,221,33,0,64,
   17,0,27,205,194,4,201
```

Una vez rodado el cargador y ejecutada la rutina con RANDOMIZE USR 50000 (pon en marcha el cassette antes de hacerlo), habrás grabado un bloque de bytes sin cabecera y con tono 21 que corresponderán a la pantalla. De ahora en adelante sólo podrás cargar esta pantalla si lo haces con la rutina de carga adecuada:

```
10 ORG       40000
20 ENT       40000
30 LD        A,#21
40 SCF
50 LD        IX,16384
60 LD        DE,6912
70 CALL      #0556
80 RET
10 CLEAR 39999: FOR
   N=40000 TO 40013: READ
   A: POKE N,A: NEXT N
20 DATA 62,21,55,221,33,0,
   64,17,0,27,205,86,5,201
```

Rueda el cargador y ejecuta la rutina (esta vez con RANDOMIZE USR 40000, no te equivoques), con lo que comprobarás que la carga es perfecta (aunque el sonido de carga es distinto).

Aparte de las dificultades a la hora de la carga que se puedan crear, si te fijas, hasta ahora hemos desechado el uso de cabeceras, pero podemos utilizarlas como un elemento más de confusión, creando las llamadas «cabeceras falsas».

Podemos crear cabeceras falsas que induzcan a un error cuando se procede a la carga desde el BASIC. Así, si grabamos los 17 bytes de una cabecera con datos que indiquen que a continuación viene un bloque de 40 Kb, pero en realidad es de 2 Kb, al proceder a la carga normal desde el BASIC se produce un error. Otra posible «trampa» sería situar la posición del comienzo de carga a partir de la 65535, generando el correspondiente error. Si deseas crear una cabecera de este tipo, puedes seguir los siguientes pasos:

— Almacenamos en memoria los datos de una cabecera que indique el tipo de programa (en este caso c/m, número 3), a continuación 10 bytes con el nombre, 2 bytes con la longitud de los datos que se pretende cargar (6912 bytes, la longitud de una pantalla), y, por fin, otros dos bytes con la dirección donde queremos cargar el programa en c/m (que será la 65535, para que produzca el error). Como ya fue explicado, los bytes 16 y 17 no influyen para nada cuando grabamos programas en c/m.

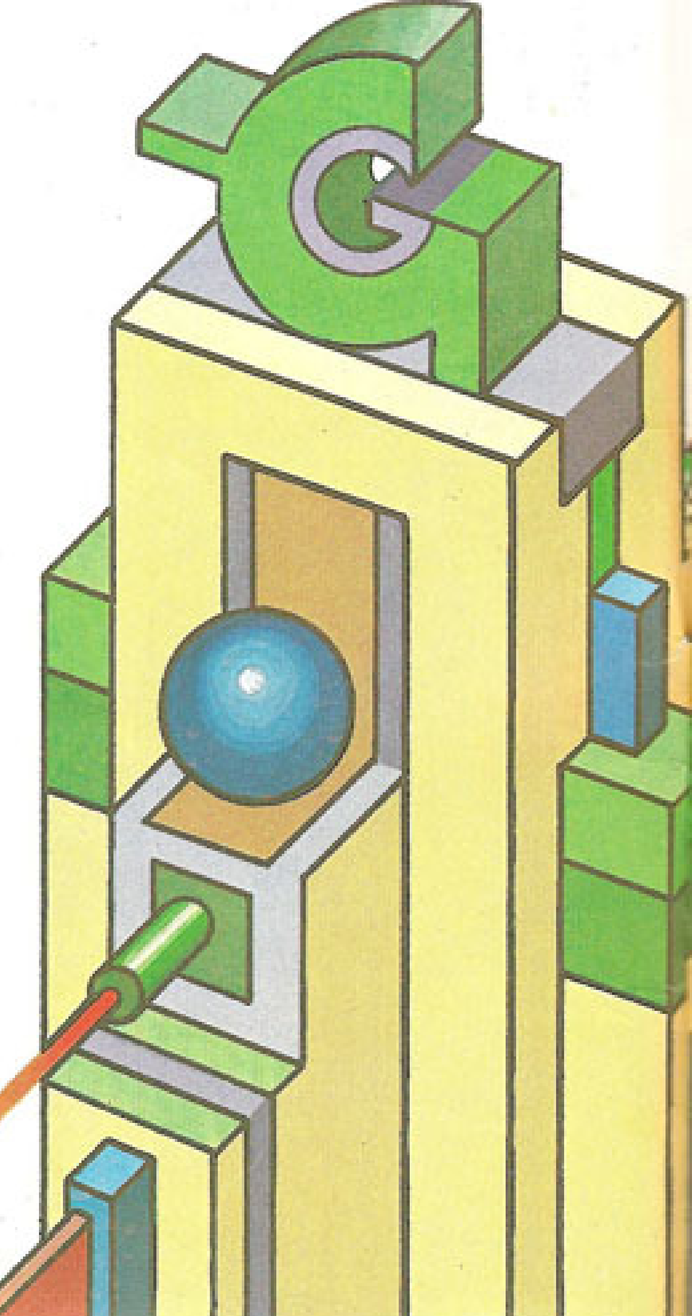
— Grabamos estos datos con el tono 0, para que sea reconocido como una cabecera. A continuación grabamos en la cinta una pantalla con tono 255.

Con estos dos pasos anteriores hemos generado una cabecera falsa que impide cualquier carga desde el BASIC causando un error evidente, aunque la pantalla real sigue en la cinta y podemos cargarla desde el c/m como hemos venido explicando hasta ahora.

ALTERANDO LOS COLORES DE CARGA

Una de las aplicaciones más vistosas que podemos conseguir manipulando las rutinas de carga es la de generar otra independiente de la primera, pero copia suya, y, alterando unos pocos bytes (ya estamos en la memoria RAM), conseguir otros colores de carga.

Como recordarás, esto mismo hicimos con las rutinas de grabación, pero en aquella ocasión había que realizar bastantes cambios. Por fortuna, la parte que se encarga de manejar los colores de la rutina, se encuentra entre las posiciones #0556 y #05E2, y, por tanto, éste es el único bloque que deberemos copiar. La acción principal de la rutina la desarrolla a su vez otra subrutina llamada por la primera, que



se extiende entre las posiciones #05E2 y #0604. Esta subrutina es la que en verdad realiza la tarea de comunicación con el exterior, detectando cuándo comienza el principio y final de la secuencia de transmisión de datos, e incluso contando los flancos de subida y bajada de los impulsos recibidos. Pero aunque sea la más importante, es utilizada constantemente mediante CALL's por la rutina controladora #0556, retornando automáticamente después de cada llamada, por lo que no importa que resida en la ROM, ya que no va a ser modificada.

Antes de nada debemos generar en la RAM una copia de la rutina que vamos a alterar, y esto podemos hacerlo de dos maneras diferentes:

— Mediante un sencillo bucle en BASIC, «PEEKeando» y «POKEando» en distintas posiciones de memoria:

```
10 CLEAR 59989
20 LET P=0
30 FOR N=1366 TO 1506
40 POKE (60000+P),PEEK N
50 LET P=P+1
60 NEXT N
```

— También, si somos impacientes y nos gusta la rapidez del código máquina, podemos hacerlo con la aplicación de la instrucción LDIR:

```
10 ORG      50000
20 ENT      50000
30 LD       HL,1366
40 LD       DE,60000
50 LD       BC,141
60 LDIR
70 RET
```

cuyo cargador BASIC será:

```
10 CLEAR 49999: FOR N=0 TO
   11: READ A: POKE
   (50000+N),A: NEXT N
20 DATA 33,86,5,17,96,234,1,
   141,0,237,176,201
```

Una vez rodado el cargador y ejecutado con RANDOMIZE USR 50000, o cuando hayamos rodado el programa que se presenta como primera opción, habremos conseguido una copia de la rutina #0556 en la posición de memoria 60000. El paso siguiente será introducir unas modificaciones:

— Como hemos venido explicando, para ejecutar la rutina de carga hemos de introducir las líneas que indican el tono, número de bytes a cargar, y dónde cargarlos:

```
10 LD       A,#FF
20 SCF
30 LD       IX,16384
40 LD       DE,6912
```

que, traducido a bytes, nos dan los números que debemos POKEar:

```
10 FOR N=59990 TO 59999:
   READ A: POKE N,A: NEXT N
20 DATA 62,255,55,221,33,0,
   64,17,0,27
```

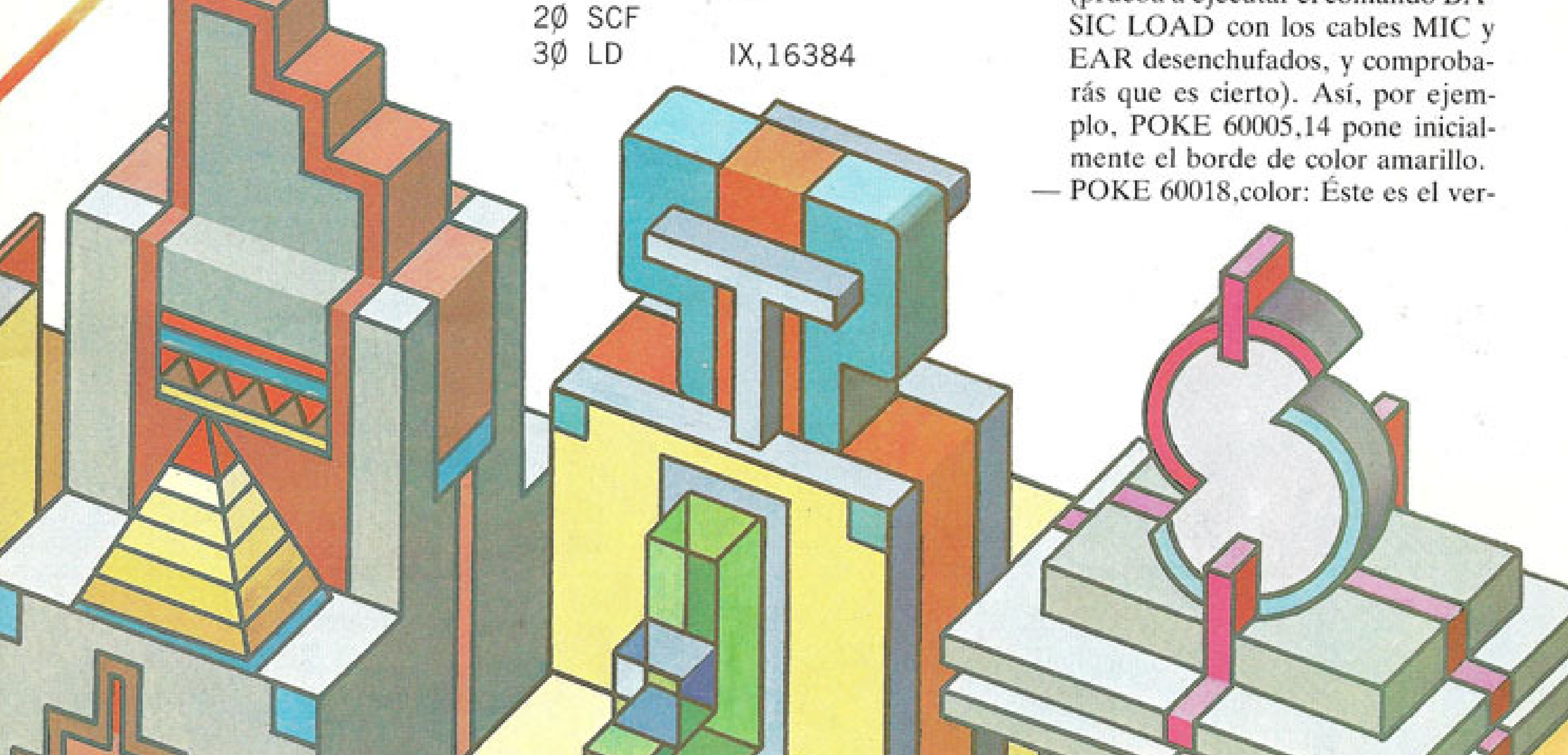
Además, la rutina original contiene un solo salto absoluto (JP #05CA), que, como es lógico, no cumple su misión en la nueva posición y sigue pasando a la ROM. Esto lo podemos evitar con POKE 60128,212: POKE 60129,234.

Hasta este momento hemos conseguido una copia exacta de la rutina de carga residente en ROM. Si ejecutas RANDOMIZE USR 59990 podrás conseguir cargar una pantalla que haya sido grabada previamente sin cabecera (o evitando ésta).

Para modificar los colores de carga deberemos realizar dos únicos POKE's que a continuación explicamos:

— POKE 60005, color+8: Si en esa dirección de memoria introducimos el código del color elegido sumándole 8, estaremos modificando la línea de código máquina LD A,#0F de la rutina original, la cual ponía de color blanco el borde antes de recibir ningún impulso del cassette (prueba a ejecutar el comando BASIC LOAD con los cables MIC y EAR desenchufados, y comprobarás que es cierto). Así, por ejemplo, POKE 60005,14 pone inicialmente el borde de color amarillo.

— POKE 60018,color: Éste es el ver-



dadero punto clave. POKEando en esta dirección un número entre 0 y 7, conseguimos cambiar el color de los dos tipos de rayas que aparecen en la rutina de carga. Inicialmente dicha posición contiene el valor 2 correspondiente al color rojo (OR #02). El único pequeño fallo, o más bien limitación existente, es que sólo podemos elegir un color y los demás son generados a partir de éste. Así conseguimos una serie de «familias»:

* POKE 60018,2 o POKE 60018,5 determina los colores usuales para la carga, es decir, el rojo y su complementario (azul claro) para las primeras líneas, y el color del borde cambiante durante la espera de llegada de datos. El amarillo y azul oscuro son seleccionados para el otro tipo de líneas.
* POKE 60018,1 o POKE 60018,6 es justo el caso contrario al anterior, azul oscuro y amarillo para las primeras y rojo y azul claro para las otras.
* POKE 60018,0 o POKE 60018,7 selecciona el color blanco y negro para las primeras líneas, y verde/magenta para las segundas.

El caso opuesto a éste sería POKE 60018,3 o POKE 60018,4.

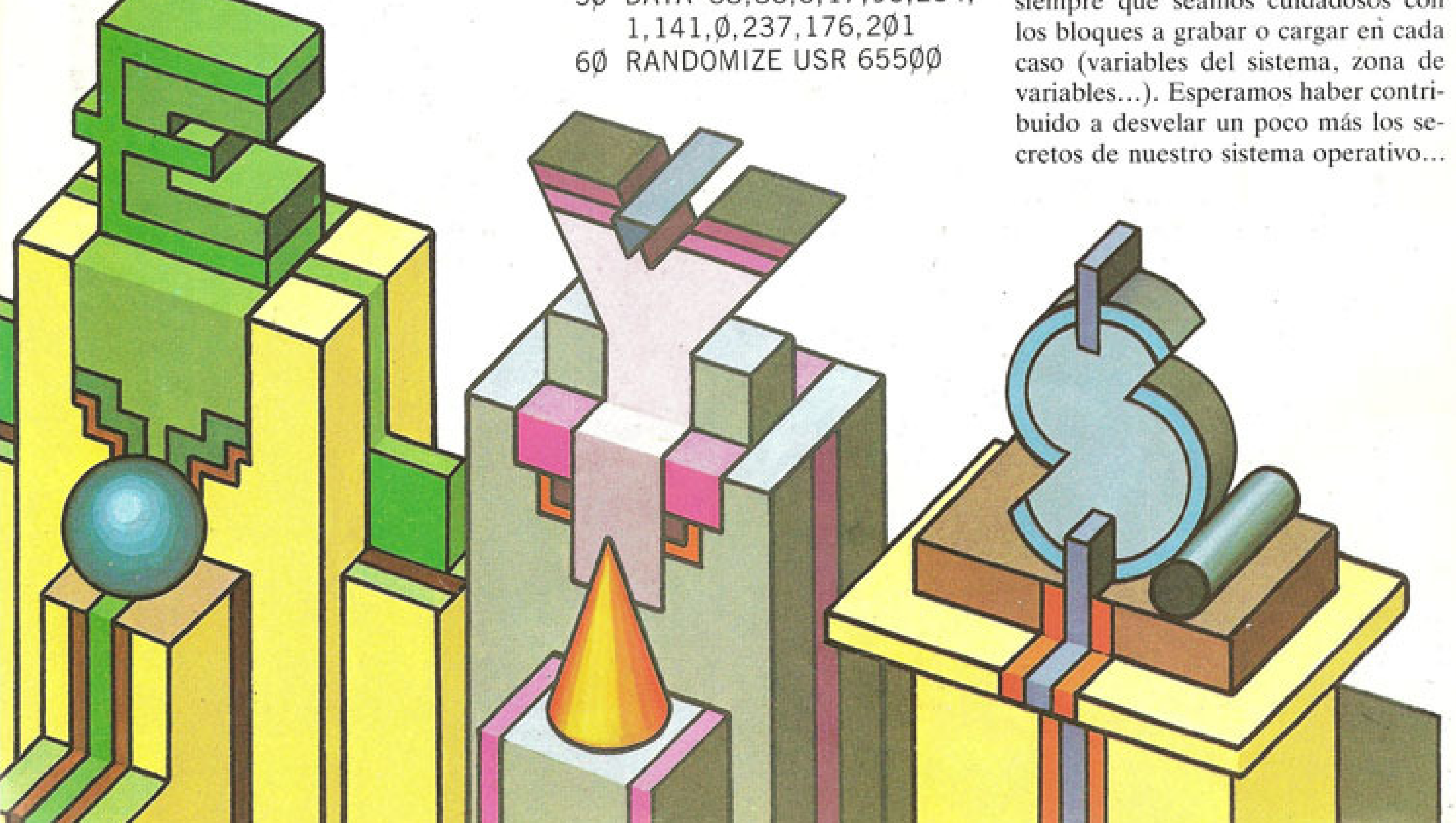
Si has seguido con paciencia todos los pasos indicados, y has realizado todos los POKE's sin equivocarte, tendrás ya diseñada tu rutina de carga con los colores que has elegido. Para ejecutarla utiliza RANDOMIZE USR 59990, y conseguirás cargar la pantalla que previamente grabaste sin cabecera.

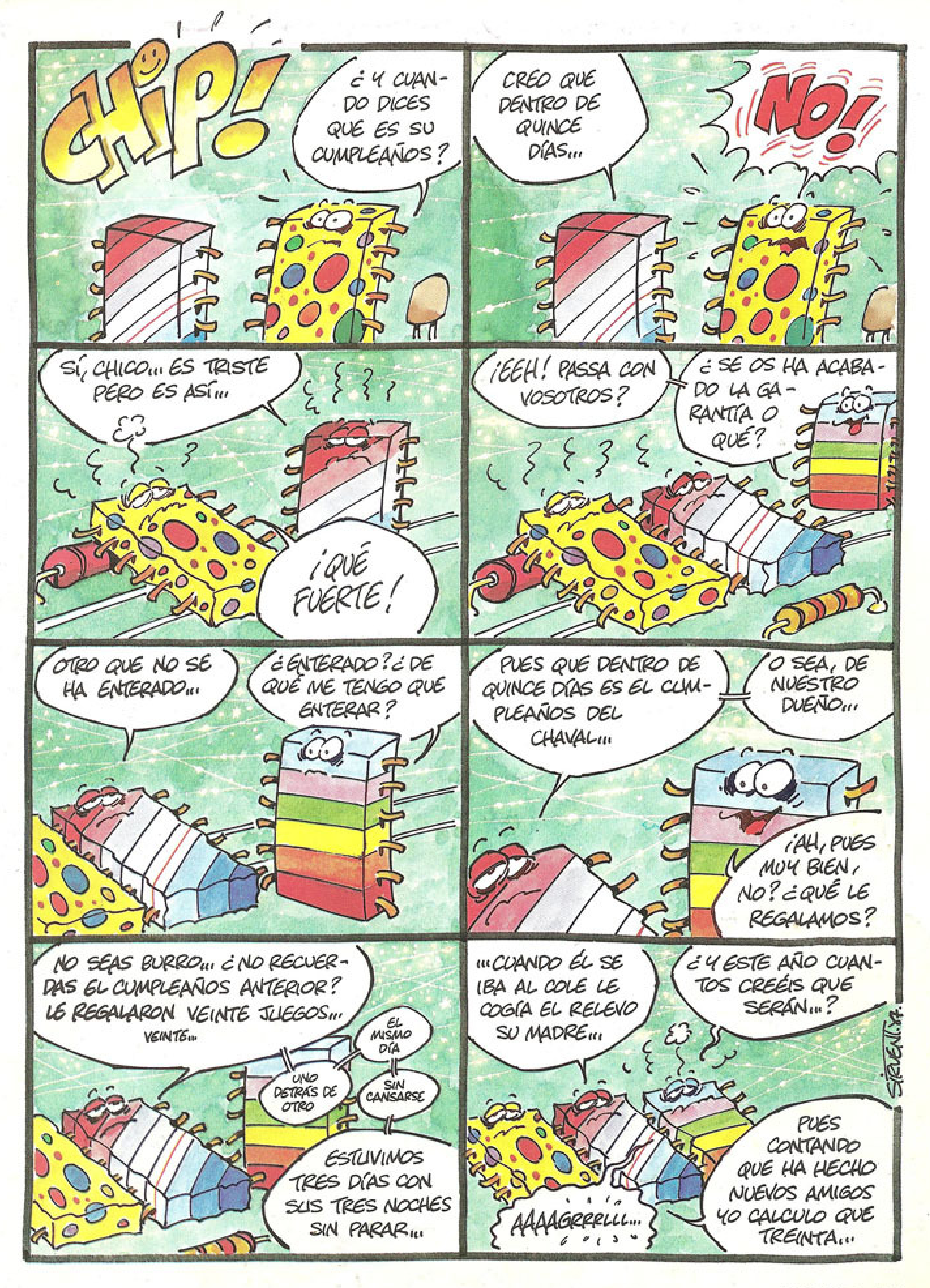
Si lo que deseas es cargar desde el código máquina cualquier pantalla grabada desde el BASIC, pero utilizando su nueva rutina, recuerda que deberás simular dos efectos de carga, uno de ellos para «eludir» la cabecera. A modo de resumen, presentamos la rutina completa incluyendo esta última modificación. Tú tan sólo deberás rodarla y dejarás perplejas a tus amistades:

```
  1 REM  ********************
  2 REM  *     MOLISOFT     *
  3 REM  ********************
 1Ø REM
 2Ø REM Carga tus pantallas con
     distintos colores
 3Ø REM
 4Ø CLEAR 59989: FOR N=Ø TO
     11: READ A: POKE
     (655ØØ+N),A: NEXT N
 5Ø DATA 33,86,5,17,96,234,
     1,141,Ø,237,176,2Ø1
 6Ø RANDOMIZE USR 655ØØ
 7Ø POKE 6Ø128,212: POKE
     6Ø129,234
 8Ø POKE 6ØØØ5,14: POKE
     6ØØ18,4
 9Ø FOR N=5999Ø TO 59999:
     READ A: POKE N,A: NEXT N
1ØØ DATA 62,Ø,55,221,33,Ø,Ø,
     17,17,Ø
11Ø RANDOMIZE USR 5999Ø
12Ø FOR N=5999Ø TO 59999:
     READ A: POKE N,A: NEXT N
13Ø DATA 62,255,55,221,33,
     Ø,64,17,Ø,27
14Ø RANDOMIZE USR 5999Ø
```

Hay que hacer constar que la rutina no es reubicable debido en gran parte a la instrucción JP, pero prestando un poco de atención se puede calcular la dirección de salto, y modificar todas las rutinas para almacenar los datos en otra posición.

Como habrás notado, todos los ejemplos han sido destinados a la manipulación de pantallas en bloque, ya que se ha considerado que, debido a su presentación visual, es lo que más fácilmente se comprende, lo cual es muy importante al tratar con datos en código máquina. Que se haya hecho así no obsta para que no podamos trabajar con cualquier bloque en código máquina o incluso programa BASIC, siempre que seamos cuidadosos con los bloques a grabar o cargar en cada caso (variables del sistema, zona de variables...). Esperamos haber contribuido a desvelar un poco más los secretos de nuestro sistema operativo...




CHIP!
¿Y CUANDO DICES QUE ES SU CUMPLEAÑOS?
CREO QUE DENTRO DE QUINCE DÍAS...
NO!
SÍ, CHICO... ES TRISTE PERO ES ASÍ...
¡QUÉ FUERTE!
¡EEH! PASSA CON VOSOTROS?
¿SE OS HA ACABADO LA GARANTÍA O QUÉ?
OTRO QUE NO SE HA ENTERADO...
¿ENTERADO? ¿DE QUÉ ME TENGO QUE ENTERAR?
PUES QUE DENTRO DE QUINCE DÍAS ES EL CUMPLEAÑOS DEL CHAVAL...
O SEA, DE NUESTRO DUEÑO...
¡AH, PUES MUY BIEN, NO? ¿QUÉ LE REGALAMOS?
NO SEAS BURRO... ¿NO RECUERDAS EL CUMPLEAÑOS ANTERIOR? LE REGALARON VEINTE JUEGOS...
VEINTE...
UNO DETRÁS DE OTRO
EL MISMO DÍA
SIN CANSARSE
ESTUVIMOS TRES DÍAS CON SUS TRES NOCHES SIN PARAR...
...CUANDO ÉL SE IBA AL COLE LE COGÍA EL RELEVO SU MADRE...
¿Y ESTE AÑO CUANTOS CREÉIS QUE SERÁN...?
AAAAGRRRLLL...
PUES CONTANDO QUE HA HECHO NUEVOS AMIGOS YO CALCULO QUE TREINTA...

GRAFICOS DEFINIDOS POR EL USUARIO (I)

Una vez has definido un juego de figuras, es muy fácil crear una gran variedad de cuadros tan sólo variando la disposición de los UDG y cambiando el fondo.

En el último artículo de la serie se indicaba la manera de que pudieses superar cualquier límite que tenga tu ordenador en el número de UDG que puedes crear, y cómo puedes definir el juego de caracteres. Este artículo es el primero de una serie de dos que te mostrarán cómo puedes ahorrar una gran cantidad de tiempo y de esfuerzo utilizando los UDG para formar un cuadro en la pantalla. UDG corresponde a las siglas de *User Defined Graphics* (Gráficos Definidos por el Usuario).

¿POR QUE HAY QUE EMPLEAR LOS UDG?

Supón que deseas crear un cuadro de la jungla —quizá una página de título para un nuevo y excitante juego. Existen básicamente dos maneras de hacerlo. Podrías dibujar con DRAW cada parte del cuadro y colorearlo como deseases. Pero supón que quisieras un bosque como parte de tu fondo. Tendrías que dibujar un tronco y tres copas para cada árbol que incluyeses en el cuadro, o sea, una tarea realmente tediosa. Y, desde luego, cada uno de estos troncos y copas serían iguales. En cambio, si puedes diseñar uno o varios UDG, que se combinen para formar un árbol, sólo tendrás que colocar con un PRINT las figuras recién creadas.

Este último procedimiento tiene la evidente ventaja de que te ahorra la necesidad de especificar cada uno de los detalles cada vez que deseas tener un árbol en la pantalla, además de otras ventajas adicionales.

Una de ellas es que ahorras tiempo. Los ordenadores presentan caracteres con PRINT mucho más rápidamente que con DRAW trazando líneas de alta resolución (lo cual también es verdad si los caracteres son definidos por el usuario y muy complicados). Por tanto, si almacenas tu cuadro en forma de UDG, cuando el ordenador llega a la parte de tu programa que presenta el cuadro, no tienes que estar aguardando un largo tiempo mientras el ordenador lleva trabajosamente tu obra a la pantalla: aparece casi instantáneamente.

LIBERTAD DE ELECCION

Otra ventaja es que puedes variar fácilmente, y cuando lo desees, el tamaño del objeto. Con el ejemplo anterior puede variar la altura del árbol simplemente añadiendo o quitando uno o más UDG del *tronco*, o alterar su follaje empleando una combinación distinta de UDG de *hoja*. Naturalmente, esto también puedes hacerlo con los comandos de alta resolución de tu ordenador, pero los UDG son mucho más cómodos. Para empezar, tratas con los números más manejables de la pantalla de baja resolución.

Una vez has definido los UDG para tu cuadro, permanecen en la memoria (a menos que los cambies), aunque no son necesaria ni permanentemente accesibles desde el teclado; si tienes varios bancos o juegos de caracteres al mismo tiempo en la memoria, tendrás que *activar* el banco que debe ser accesible desde el teclado. Para esto bastará simplemente con variar el puntero de los UDG.

Esto significa que podrás utilizar las figuras que has diseñado tantas veces como desees en tus programas: el único límite es la memoria del ordenador. En consecuencia, en tu pro-



- COMO DECIDIR CUANDO HAY QUE UTILIZAR LOS UDG
- FORMACION DE UN CUADRO CON VARIOS CARACTERES
- UN CUADRO DE LA JUNGLA
- UN COCODRILO, UN ELEFANTE Y ARBOLES
- DIBUJO DEL FONDO
- MEZCLA DE GRAFICOS DE ALTA Y DE BAJA RESOLUCION

grama resultará tan fácil tener un cuadro de bosque como un solo árbol.

No obstante, el empleo de los UDG tiene algunos pequeños inconvenientes. Para empezar, tienes que definir todas las figuras y teclear los datos en el ordenador, aunque normalmente, esto no es peor que trabajar con gráficos de alta resolución. Pero también hay que utilizar una parte de la limitada memoria de tu ordenador para almacenar los UDG y, a menos que seas muy cuidadoso, acabarás por almacenarlos dos veces. En el segundo artículo de esta serie se indica cómo puedes evitar esta repetición de memoria.

¿QUE IMAGENES EMPLEAN LOS UDG?

Por las razones expuestas anteriormente, hay una serie de imágenes que puedes dibujar de manera mucho más eficiente con UDG que otras.

Como regla general práctica, si el cuadro tiene un número relativamente pequeño de objetos macizos en partes que aparecen con una forma similar varias veces, o si deseas emplear una imagen varias ocasiones durante la ejecución de un programa, podrás ahorrar tiempo y esfuerzo utilizando los UDG.

Pero si deseas tener un cuadro muy detallado que sólo vas a emplear una vez en el programa, probablemente



será mejor que utilices los comandos de alta resolución del ordenador.

Por ejemplo, si el cuadro debe tener como telón de fondo una pared de ladrillos, la podrás formar rápidamente empleando sencillamente uno o dos UDG que presentarás con PRINT repetidamente en la parte del cuadro en la que desees que aparezca la pared. Una alternativa podría ser dibujar una serie de líneas en una zona coloreada para simular las juntas de los ladrillos.

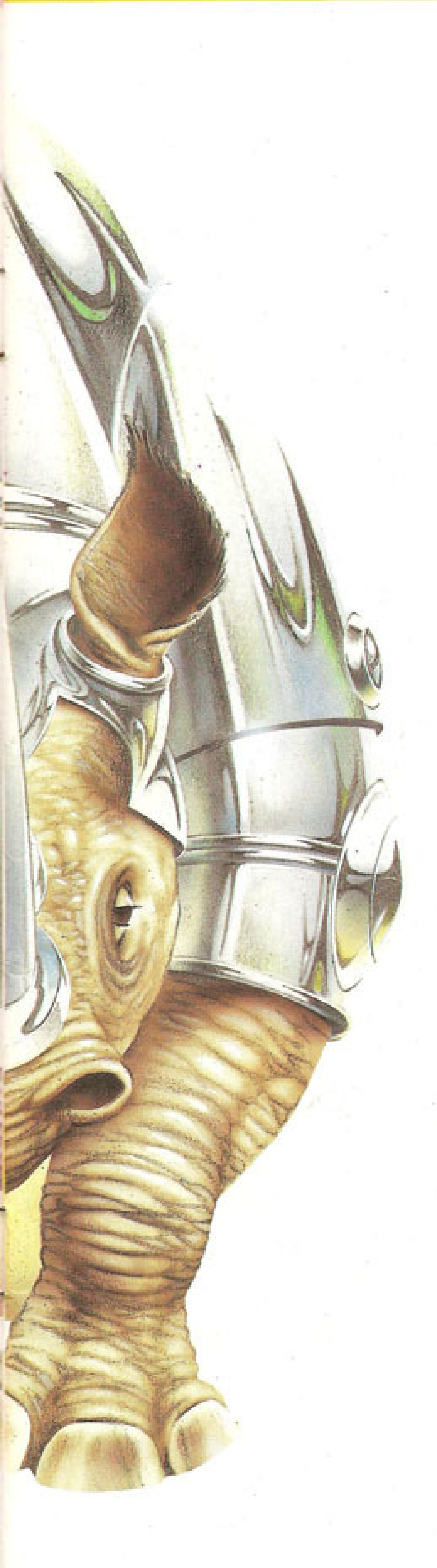
Otro ejemplo sería el cuadro de la jungla mencionado anteriormente. Ya se ha indicado que puedes ahorrar tiempo definiendo los UDG para los árboles que debe contener cualquier escenario de jungla. Los animales que deben acompañar a los árboles también son muy adecuados para ser definidos con UDG, puesto que así podrás emplearlos en cualquier punto del programa, para animarlos, o para disponer de un gran número de los mismos.

UN CUADRO DE LA JUNGLA

A continuación se presenta un programa que sirve para definir los caracteres típicos que probablemente desearás incluir en la escena de la jungla.

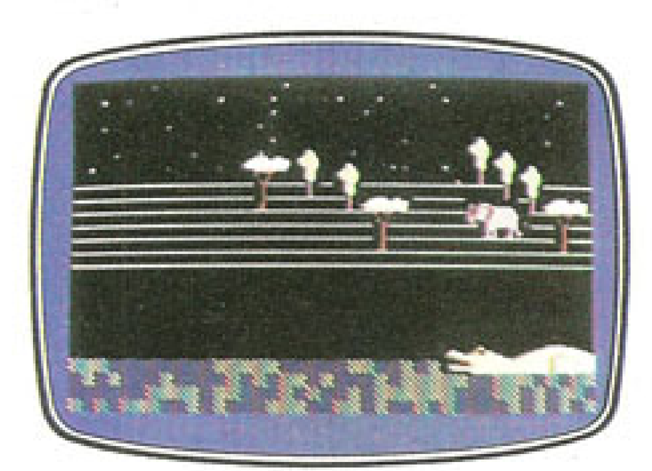
```
10 CLEAR 63500
100 REM Pokear datas del
     cocodrilo
110 POKE 23676,255
120 FOR n=USR "a" TO USR
     "r"+7: READ a: POKE n,a:
     NEXT n
250 REM Pokear datas del
     elefante
260 POKE 23676,249
270 FOR n=USR "a" TO USR
     "m"+7: READ a: POKE n,a:
     NEXT n
280 REM Pokear datas de los
     arboles
290 POKE 23676,248
300 FOR n=USR "a" TO USR
     "o"+7: READ a: POKE n,a:
     NEXT n
400 REM
410 BORDER 1: PAPER 8: CLS
420 FOR n=1 TO 8: PRINT
```





```
     PAPER 5;" ";TAB 31;" ":
     NEXT n
430  FOR n=1 TO 14: PRINT
     PAPER 4;" ";TAB 31;" ":
     NEXT n
440  PLOT 0,110: DRAW 142,
     -100,-PI/3: PLOT 160,
     110: DRAW -60,-42,PI/3
450  REM Print cocodrilo
460  POKE 23676,255: INK 2
470  PRINT AT 19,20;: FOR
     n=144 TO 155: PRINT
     CHR$ n;: NEXT n
480  PRINT AT 20,20;CHR$ 156;
     CHR$ 157;CHR$ 158;CHR$
     159;CHR$ 159;CHR$ 159;
     CHR$ 159;CHR$ 159;CHR$
     160;CHR$ 159;CHR$ 159;
     CHR$ 159
490  FOR n=0 TO 31: PRINT INK
     1; PAPER 4;CHR$ 161;:
     NEXT n
750  REM Print elefante
760  INK 7
770  POKE 23676,249
780  PRINT AT 8,9;CHR$ 144;
     CHR$ 145;CHR$ 146;CHR$
     147;AT 9,9;CHR$ 148;
     CHR$ 149;CHR$ 150;CHR$
     151;CHR$ 151;CHR$ 152;
     AT 10,10;CHR$ 153;CHR$
     154;CHR$ 155;CHR$ 156
790  REM Print arboles
800  POKE 23676,248
810  LET x=6: LET y=14: GO
     SUB 840: LET x=5: LET
     y=18: GO SUB 840
820  LET x=4: LET y=0: GO SUB
     845: LET x=4: LET y=3: GO
     SUB 845
830  GO TO 850
840  PRINT INK 4;AT x,y;CHR$
     151;CHR$ 152;CHR$ 153;
     CHR$ 154; INK 2;AT x+1,
     y+1;CHR$ 155;CHR$ 156;
     AT x+2,y+1;CHR$ 157;AT
     x+3,y+1;CHR$ 157;AT
     x+4,y+1;CHR$ 157;AT x
     +5,y+1;CHR$ 158: RETURN
845  PRINT INK 4;AT x,y;CHR$
     144;CHR$ 145;AT x+1,y;
     CHR$ 146;CHR$ 147;AT
     x+2,y;CHR$ 148;CHR$
     149; INK 2;AT x+3,y+1;
     CHR$ 150;AT x+4,y+1;
     CHR$ 150;AT x+5,y+1;
     CHR$ 150: RETURN
970  RETURN
1300 REM cocodrilo
1310 DATA 0,0,1,7,15,15,9,5,
     0,0,128,192,248,255,
     127,95,1,3,6,12
1320 DATA 62,255,255,255,
     192,224,176,159,191,
     255,255,255
1330 DATA 0,0,0,0,0,248,252,
     255,0,0,0,0,0,0,1,207,0,
     0,0,1,15,127,255,255
1340 DATA 0,3,63,255,255,
     255,255,255,127,255,
     255,255,255,254,249,
     247,248
1350 DATA 255, 255, 255,255,
     15,255,255
1360 DATA 0,224,254,255,
     255,255,255,255,0,0,0,
     192,248,255,255,255,0,
     2,4,7,7,
1370 DATA 3,0,0,21,1,164,73,
     255,255,0,0
1380 DATA 255,127,63,63,
     255,255,127,31,255,
     255,255,255,255,255,
     255,255
1390 DATA 239,239,239,239,
     239,247,247,247,60,
     255,255,255,255,255,
     255,255
1780 REM elefante
1790 DATA 0,0,0,8,28,25,51,
     51,0,0,0,126,255,193,
     253,253,0,0,0,0,0,255,
     255
1800 DATA 255,0,0,0,0,0,248,
     252,254
1810 DATA 102,111,111,111,
     125,57,26,0,253,251,
     251,251,231,31,15,15,
     255,255
1820 DATA 255,255,255,255,
     255,255,254,255,255,
     255,255,255,255,255
1830 DATA 0,0,128,64,32,16,
     12,0,15,15,15,14,14,14,
     14,31,255,240,224,224
1840 DATA 224,224,224,224,
```

```
      255,63,59,59,57,57,121,
      248
185Ø  DATA Ø,Ø,128,192,224,
      224,128,Ø
186Ø  REM arbol 1
187Ø  DATA Ø,Ø,Ø,Ø,1,1,3,7,Ø,
      Ø,Ø,Ø,224,24Ø,248,248,
      15,63,63,63,31,15,3,3
188Ø  DATA 252,254,254,254,
      254,254,252,252
189Ø  DATA 3,3,3,3,3,1,Ø,Ø,
      248,248,248,248,248,
      248,24Ø,96,96,96,96,96,
      96,96
19ØØ  DATA 96,96
191Ø  REM arbol 2
192Ø  DATA Ø,3,15,31,127,127,
      63,1,7,15,255,255,255,
      255,255,255,15,63,255,
      255
193Ø  DATA 255,255,255,255,
      Ø,128,248,248,248,248,
      24Ø,224
194Ø  DATA 255,227,96,48,24,
      25,13,15,252,24Ø,96,96,
      192,192,128,128,7,7,7,
      7,7
195Ø  DATA 7,7,7,7,7,7,15,15,
      15,31,63
```



Si introduces y haces correr el programa en el ordenador, podrás ver la clase de cuadro que podrás crear con UDG. No te preocupes si la mitad superior de la pantalla aparece algo vacía en este momento, ya que en el próximo artículo de esta serie se terminará el cuadro.

Podrás comprobar la versatilidad de los UDG observando el agua que hay debajo del cocodrilo. En realidad es un solo UDG que se repite en toda la zona del agua.

También podrás observar que el cuadro no sólo contiene UDG, sino también algunas líneas de alta resolución que dan la impresión de que en el cuadro también hay colinas.

En este ejemplo puedes comprobar que si bien los UDG tienen muchas ventajas con respecto a los comandos

de alta resolución, a menudo podrás obtener muy buenos resultados empleando ambos procedimientos en el mismo cuadro.

En el siguiente artículo de esta serie se añadirán más cosas al cuadro. Por tanto, salva esta parte en la cinta para evitar tener que escribirlo otra vez.

COMO FUNCIONA EL PROGRAMA

El programa se ha dividido en dos partes. En la primera parte se definen todos los caracteres y, en la segunda, los UDG se presentan en las posiciones adecuadas.

El programa introduce los datos para cada animal y árbol. Una vez definidos los UDG, el programa presenta el cuadro en la pantalla.

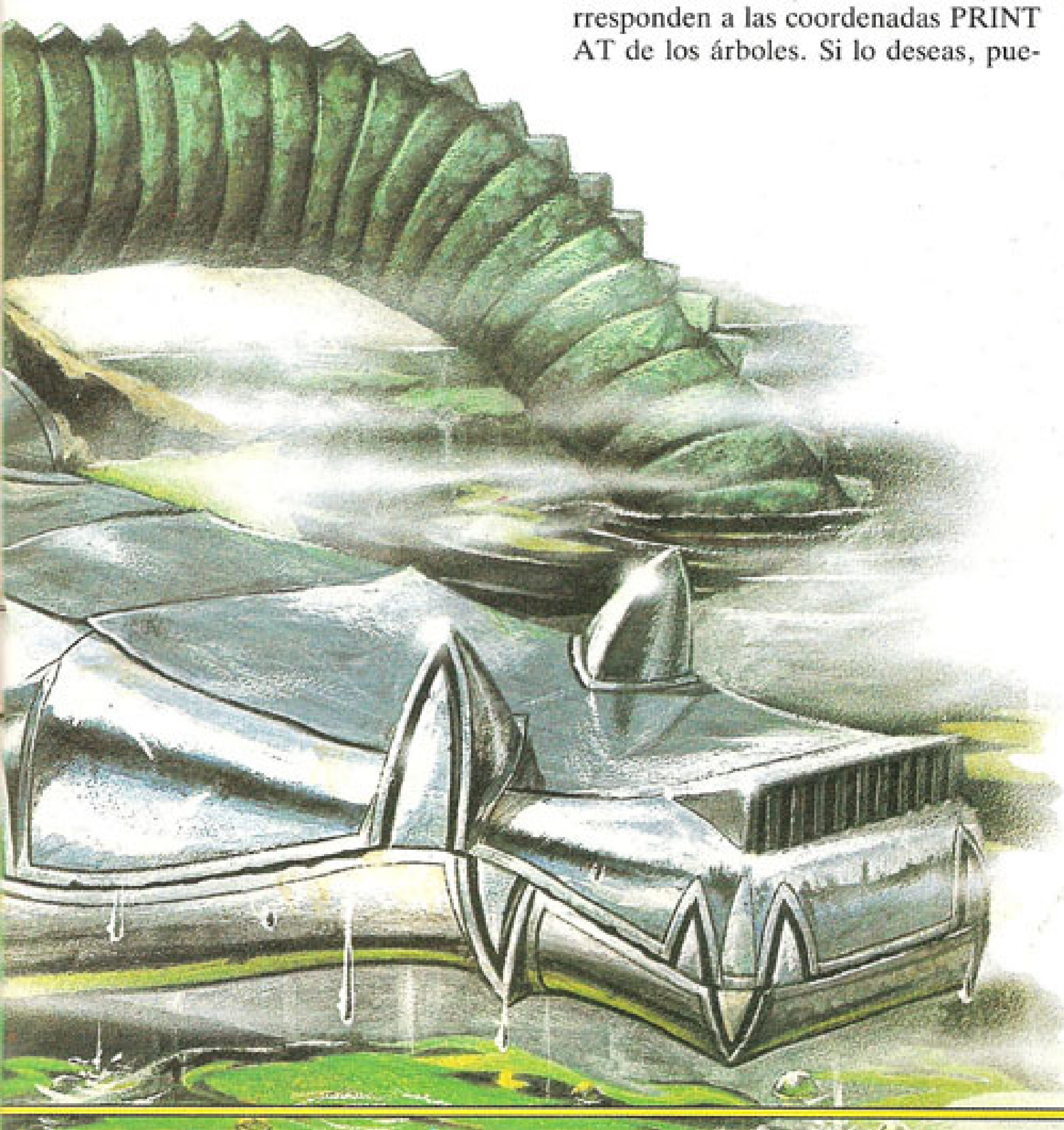
Para ello, el ordenador emplea una serie de PRINT AT, presentando cada animal y los árboles al mismo tiempo. También emplea comandos de color locales para producir el cuadro en colores que podrás ver cuando hagas correr el programa.

Un comando de color sólo es válido para la sentencia que se encuentra en la línea de programa. El PAPER de color general se fija a 8, que es transparente. Esto deja PAPER igual como estaba, con lo que se asegura que el fondo no quede estropeado por los UDG presentados.

FONDO MULTICOLOR

El fondo es establecido con las líneas 410 a 440. La primera fija el color, y la segunda presenta el cielo cyan y el suelo verde. Los distintos colores de PAPER se consiguen imprimiendo espacios en el color PAPER correspondiente, utilizando dos bucles FOR...NEXT.

Los árboles se imprimen con subrutinas (líneas 810 y 820), puesto que en el cuadro hay varios tipos de árboles. Las variables x e y de estas líneas corresponden a las coordenadas PRINT AT de los árboles. Si lo deseas, puedes añadir fácilmente más árboles al cuadro empleando más coordenadas y añadir más GOSUB... a estas dos líneas.



Una vez introducidos estos UDG en la memoria, los podrás emplear en tus programas cuantas veces lo desees, y también podrás hacer modificaciones en el cuadro hasta que tenga exactamente el aspecto que desees.

Es posible que te guste sustituir el elefante único que se presenta en este momento por toda una manada. Esto podrás hacerlo de manera parecida al método empleado para presentar varios árboles: con un bucle FOR...NEXT o con varios GOSUB.

En la siguiente parte de este artículo se verá como puedes añadir más figuras a la escena, así como aplicar alguna animación a la misma.

Deberás tener mucho cuidado al introducir los datos, puesto que es inevitable que haya una gran cantidad.

Si tu ordenador se detiene al hacer correr el programa con un mensaje de error OUT OF DATA, significa que no hay suficientes datos.

Si has indicado al ordenador que lea la cantidad correcta de datos, pueden haber dos posibles razones para ello: puedes haber olvidado uno o más números o puedes haber introducido puntos en lugar de comas. Un punto en lugar de una coma cambia dos números en uno, puesto que el punto se interpreta como un punto decimal.

La única solución para estos dos problemas es comprobar, y volver a comprobar, los datos hasta que encuentres el error. Resulta de mucha ayuda incluir un comando PRINT en el bucle que introduce los datos, con el que el ordenador presenta cada número al introducirlo. De esta manera podrás comprobar cada número a medida que se introduce.

Si has introducido demasiados datos, o la cantidad correcta pero erróneos, el programa correrá sin ninguna dificultad, pero verás una serie muy extraña de imágenes: por ejemplo, el cocodrilo podrá tener una trompa o el elefante podrá parecerse a una copa de árbol. Como antes, la única solución consiste en comprobar todos los datos introducidos.

AZAR Y PROBABILIDAD

Tanto si estás interesado en estrategias de juegos como en predecir situaciones de vida o muerte, no puedes esperarlo todo de la suerte. Analiza mejor cómo puedes obtener información sobre el futuro, y gana.

La fuerza de un ordenador reside en su capacidad para obedecer instrucciones de una forma repetitiva, precisa y rápida. Si lo comparamos con el proceso casi instantáneo del cerebro humano, la actuación de un ordenador es desde luego insignificante. El cerebro humano es único para hacer juicios y comparaciones de parámetros como la distancia, la velocidad o la intensidad de la luz.

Pero también estas funciones más sutiles fallan estrepitosamente cuando tratan de adivinar el resultado de un suceso. Y, sin embargo, es importante poder afirmar con seguridad que «a efectos de un seguro, se considera que una persona llega a vivir hasta los 65 o los 70 años», o bien que «no es probable que se dé un terremoto de gran alcance en Inglaterra en lo que queda de siglo». Tales afirmaciones son normales en el lenguaje de la vida diaria (ponderamos el riesgo) y son también importantes para fines científicos, sociales o comerciales. Cuando utilizamos términos como *riesgo, pronóstico, duda, esperanza* y *posibilidad* estamos haciendo cálculos mentales de *probabilidades*.

LA PROBABILIDAD TAL COMO ES

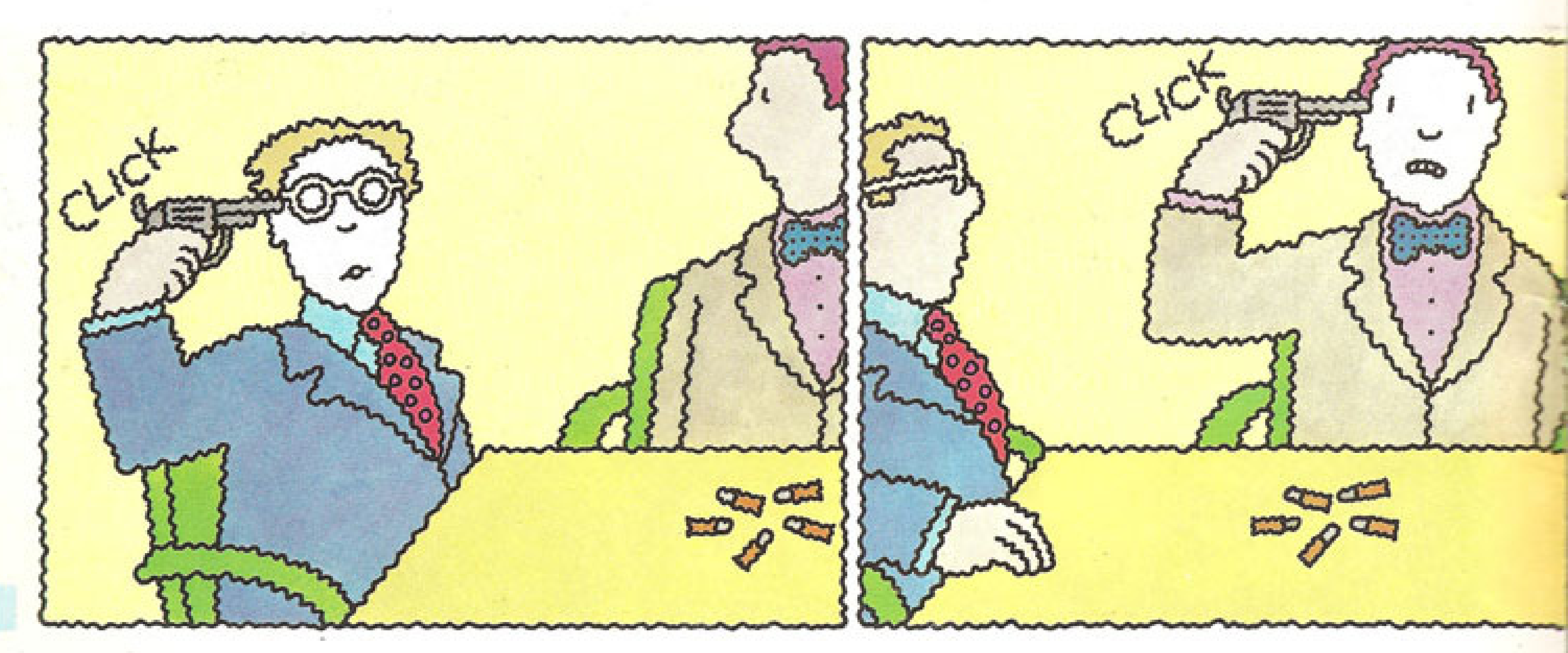
La probabilidad es la medida científica del azar, y se emplea para juzgar el posible resultado de un suceso. Se basa en la existencia de un número finito de resultados posibles, como pueden darse en los partidos de fútbol, el lanzamiento de una moneda, tirada de dados, reparto de cartas de baraja o el juego de las máquinas de frutas. Está claro que podemos medir o cuantificar los resultados, de forma que sucesos como las carreras de caballos o una competición de baloncesto constituyen la materia difícil de la probabilidad. Si tú dices «*Espero* ganar», lo que estás diciendo es que existe una elevada *probabilidad* de que te sonría la suerte.

Muchos se fían sólo en la intuición, y ésta es su principal herramienta de cálculo de probabilidades. Pronostican, y basta. Pero tú puedes hacerte una idea más precisa de ciertas situaciones en las que interviene el azar si te detienes a examinar cuáles son los sucesos posibles, y aunque nunca tendrás la certeza absoluta estarás en mejores condiciones que el que emite su pronóstico poco o nada *documentado*.

PROBABILIDAD E INFORMATICA

¿Y qué tienen en común la probabilidad y la informática? Pues, hablando en castizo, *la tira*. Aunque el tipo de probabilidad antes mencionada es bastante amplia y depende de muchos factores, es posible deducir reglas matemáticas para ciertos tipos de sucesos que nos permitan predecir el resultado más verosímil con un cierto margen de seguridad.

Los ordenadores pueden ser útiles por dos vertientes. Por un lado, pueden utilizarse para simular el resultado mismo, es mucho más fácil programar un ordenador para que te tire un dado



- ¿QUE ES PROBABILIDAD?
- AZAR, PROBABILIDAD Y FRECUENCIA
- PROGRAMA PARA LANZAR UNA MONEDA
- PROBABILIDAD DE VARIOS RESULTADOS
- TRIANGULO DE PASCAL
- DISTRIBUCION DE FRECUENCIAS
- PREDICCION DE RESULTADOS

dos mil veces que hacerlo tú con tu propia mano. Por otro lado, si sabes las fórmulas del resultado esperado, puedes emplear el ordenador para que te calcule también el resultado.

Según sean tus intereses, esto puede reducirse a un mero ejercicio teórico, o convertirse en la base de muchas aplicaciones prácticas. Considera sólo el mundo de los juegos de azar, en los que, por ejemplo, tu apuesta depende de la verosimilitud de determinados resultados. Pero también se podría escribir un programa para determinar la probabilidad de que llueva en un día determinado (¡o la probabilidad de que acontezca una erupción volcánica en la península!) De momento ciñámonos a la teoría. Más adelante INPUT te enseñará a montar algunas de esas aplicaciones prácticas.

MEDIR LA PROBABILIDAD

La teoría de la medición de la probabilidad pide que conozcas de antemano el número de sucesos posibles de un determinado experimento, y la frecuencia con que suelen ocurrir cada uno de ellos. La probabilidad de que ocurra un determinado suceso es igual al número de veces que suele suceder (su frecuencia) partido por el número total de sucesos posibles.

Si un suceso es seguro de que ocurrirá, su probabilidad será, según lo dicho, igual a 1. Está claro que si siempre ocurre, su frecuencia es igual al número de resultados posibles, es decir, dividimos un número por sí mismo, luego es igual a 1. Conclusión: la probabilidad más alta de un suceso vale 1. Y para muchos sucesos posibles e independientes, si sumamos sus respectivas probabilidades nos dará también 1.

Uno de los métodos más sencillos y antiguos de razonamiento probabilístico consistía en lanzar una moneda al aire y predecir su resultado. Dado que una moneda sólo ofrece dos resultados, intuitivamente podemos deducir que si la lanzamos al aire un buen número de veces, obtendremos cara la mitad de las veces y cruz la otra mitad, dando por despreciable la bajísima probabilidad de que la moneda se quede de canto. Para ilustrar este método, introduce y ejecuta este primer programa.

```
5 BORDER 7: PAPER 7: INK 9:
  CLS
10 DIM n(4)
20 RESTORE 9000: FOR n=1 TO
   4: READ n(n): NEXT n
30 INPUT "Que test deseas (1-
   4)?",x: CLS
40 BORDER x: GO TO n(x)
50 REM Probabilidad
60 REM Lanzamiento de moneda
70 LET h=0: LET t=0
80 PRINT AT 2,5;"Pulsa SPACE
   para tirar"
90 PRINT AT 20,11;"CARA:- 0";
   AT 21,11;"CRUZ:- 0"
100 IF INKEY$<>CHR$ 32
    THEN GO TO 100
110 IF x=2 THEN FOR n=1 TO
    100
120 IF INT (RND*2)=1 THEN
    LET h=h+1: PRINT AT 10,
    13;"CARA";AT 20,18;h: GO
    TO 130
125 LET t=t+1: PRINT AT 12,
```



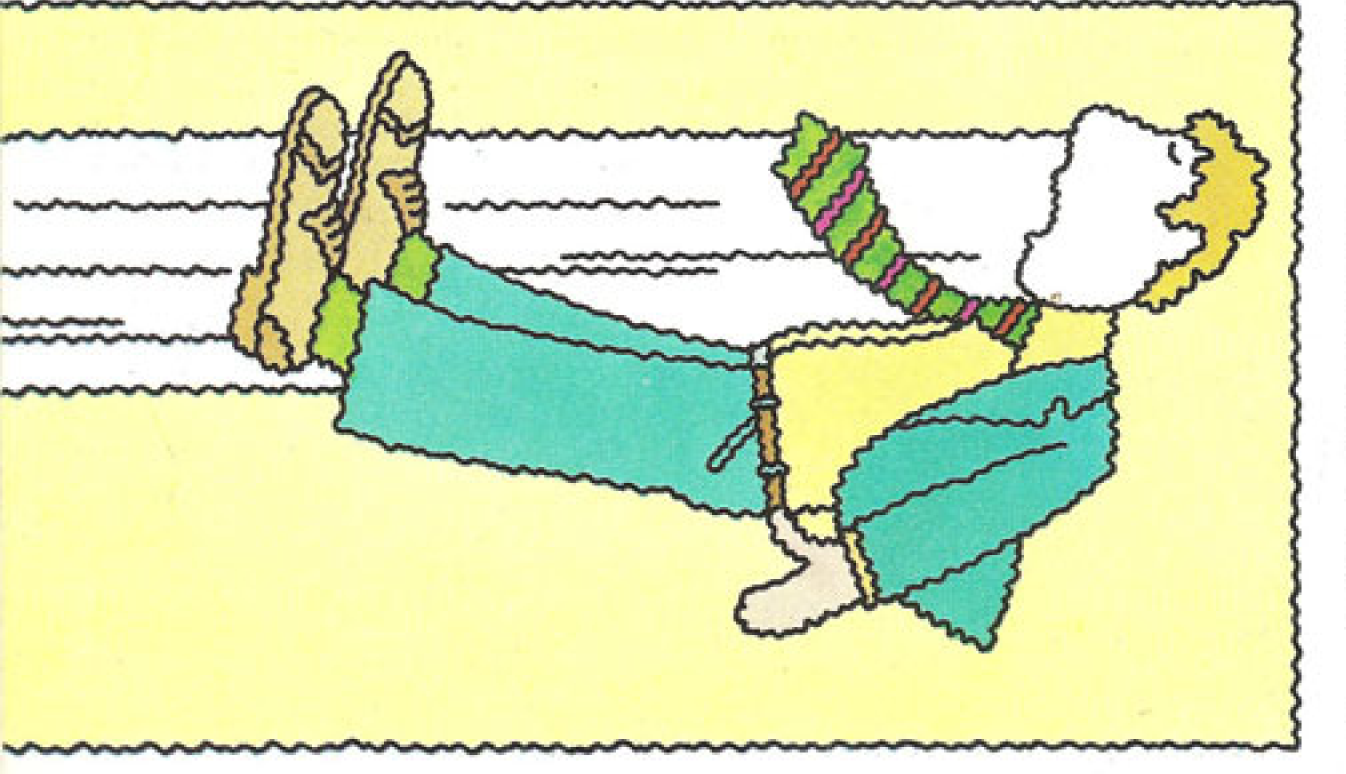
```
      13;"CRUZ";AT 21,18;t
130 IF x=1 THEN IF
      INKEY$<>CHR$ 32 THEN
      GO TO 130
140 PRINT AT 10,13;"    ";AT
      12,13;"
150 IF x=1 THEN FOR m=1 TO
      100: NEXT m: GO TO 120
155 NEXT n: STOP
9000 DATA 70,70,170,460
```

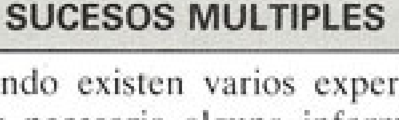
Este programa será desarrollado a lo largo del artículo. Cuando lo ejecutes (RUN) habrá que introducir un número para seleccionar una prueba. En este momento sólo has entrado la primera de las pruebas, luego escribe un 1, estás ahora en condiciones de lanzar la moneda apretando la barra espaciadora o SPACE. La esencia del programa está en la línea 120 que proporciona unos (Caras) y ceros (Cruces) al azar. Cuando sale cara, la línea 120 imprime una H, y cuando sale cruz imprime (PRINT) una T. Esta misma línea tiene un contador del número de caras y cruces que van saliendo en los distintos lanzamientos. La línea 150 establece un retardo entre un lanzamiento y otro.

Con un número escaso de lanzamientos obtendremos valores muy distintos de H y de T, pero si aumentamos éstos veremos cómo tales valores se van aproximando cada vez más, hasta distribuirse por igual la mitad de los lanzamientos: o sea, el 50 % de las veces salió Cara y el 50 %, Cruz.

Compruébalo ejecutando el programa después de haber entrado el 2, para seleccionar la segunda prueba. Esta vez, cuando aprietes la barra o SPACE, la línea 110 establece un bucle con el que se lanzará 100 veces la moneda. Verás que en pantalla sale para H y para T un número muy próximo a 50. Si quieres, cambia el 100 de la línea 110 por el 1000 y vuelve a ejecutar el programa, introduciendo como antes el número 2 para que repita la segunda prueba: resultado H y T reciben casi 500 por igual.

Desde luego, es posible obtener cara en todos los lanzamientos, nadie puede negar esto, pero es probable obtener cara sólo en la mitad de dichos lanzamientos. Anótate esto bien cuando examines más de un experimento. Mucha gente cree que si seguimos lanzando una moneda después de haber obtenido diez caras, es mayor la probabilidad de obtener cruz ahora que al principio. Pero esto no es verdad. Los sucesos pasados no influyen sobre el suceso futuro de obtener una cruz o una cara en el siguiente lanzamiento. Pero si vas a lanzar 11 monedas de golpe, deberás saber que la probabilidad de obtener 11 caras es más pequeña que la de obtener diez caras y una cruz, y es todavía más probable que obtengas un número casi igual de caras que de cruces.





SUCESOS MULTIPLES

Cuando existen varios experimentos, es necesaria alguna información más para poder predecir la probabilidad de cada suceso. Un dato informativo esencial es el total de todos los sucesos posibles. Por ejemplo, si lanzas dos veces una moneda, tres son los sucesos posibles: dos caras, cara y cruz, dos cruces. Podemos pensar que cada uno de estos sucesos puede ocurrir la tercera parte de las veces. De hecho, las probabilidades son: dos caras (1/4), dos cruces (1/4) y cara y cruz (1/2). Para entender esta última probabilidad de 1/2 deberás utilizar otra información más: el número de ocurrencias de cada suceso. Cara y cruz ocurre dos veces, ya que hay *dos* maneras de obtener este resultado (cara y cruz, o bien cruz y cara) con lo que tenemos un total de cuatro sucesos, de los cuales tres son distintos.

En la práctica, hay dos trucos matemáticos que te ahorran el esfuerzo de determinar el número de sucesos. Son el teorema del binomio y el triángulo de Pascal. Binomio significa de dos términos. Si un experimento sólo tiene dos sucesos posibles y conoces la probabilidad de cada uno de ellos, nos serviremos del teorema del binomio para obtener las probabilidades.

El teorema del binomio nos indica lo que se debe esperar de las pruebas que se repiten de un experimento con dos sucesos. Llamemos P la probabilid ad de un resultado y Q la del otro resultado (nota que P+1 vale 1, como ya sabes). Y llamemos por último N al número de sucesos.

En el ejemplo de lanzar una moneda, P es la probabilidad de que salga cara, y Q de que salga cruz. Entonces, para un lanzamiento, P y Q valen 1/2. Según el teorema del binomio, la probabilidad en un experimento que se da dos veces, es la probabilidad cuando el experimento sólo se da una vez multiplicada por sí misma. En general, la regla dice que la probabilidad hay que elevarla a N. Así para dos caras en dos lanzamientos tendremos

$P \uparrow N = \frac{1}{2} * \frac{1}{2} = \frac{1}{4}$. Hay una posibili-

dad sobre cuatro de obtener dos caras a la vez. Semejantemente, la probabilidad de obtener cinco caras a la vez, será $P \uparrow N$, o sea $\frac{1}{2} \uparrow 5$, es decir, $\frac{1}{32}$.

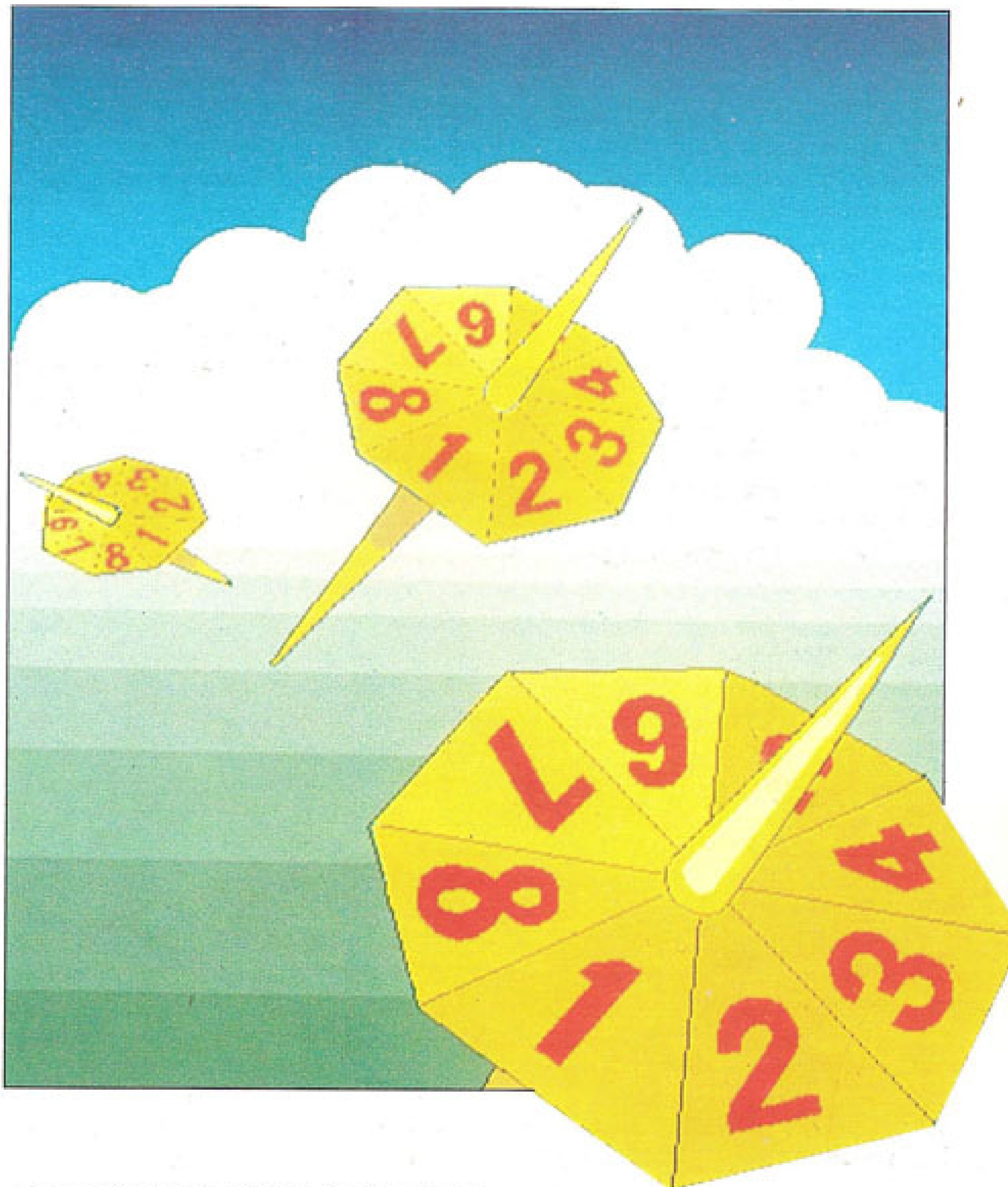
Como verás más adelante, este método se puede emplear para calcular la probabilidad de cualquier suceso cuando sólo existen dos resultados posibles, los sucesos si/no o bien cara/cruz. Pero ¿y el caso de obtener tres caras y dos cruces después de cinco lanzamientos de una sola moneda? Para responder a esto necesitamos un planteamiento más complejo.

El triángulo de números ideado por el matemático francés Blas Pascal tiene muchas aplicaciones matemáticas, y entre ellas la que nos ocupa. Ofrece todos los resultados posibles de cualquier experimento con dos resultados, y puede ser dispuesto como una sucesión de filas de números. Las primeras siete filas son:

Fila 0	1
Fila 1	1 1
Fila 2	1 2 1
Fila 3	1 3 3 1
Fila 4	1 4 6 4 1
Fila 5	1 5 10 10 5 1
Fila 6	1 6 15 20 15 6 1

Para construir un triángulo como éste, escribe primero las dos primeras filas (Fila 0 y Fila 1) que son fáciles de recordar. La Fila 2 comienza con un 1 a la izquierda y acaba con otro 1 a la derecha de la Fila 1. El número del medio (2) se obtiene sumando los números de la fila anterior (1+1). Igualmente, la Fila 3 se obtiene colocando sendos unos a la derecha y a la izquierda, y los restantes números son la suma de los números de la fila anterior (Fila 2): 1+2 y 2+1. Así se han obtenido las demás filas, como la Fila 6: 1 - 1+5 - 5+20 - 10+10 - 10+5 - 5+1 - 1. Y así podrías enriquecer este triángulo con todas las filas que desees, aunque el triángulo te ocupará entonces mucho espacio.

El triángulo de Pascal (también llamado de Tartaglia), te ofrece la información que necesitas, cuando se trata del lanzamiento de varias monedas al mismo tiempo (o de una moneda que se lanza varias veces). El número de monedas nos dice la fila que hay que inspeccionar; el número de elementos de esa fila indica el número de resultados. Por ejemplo, sólo hay dos resultados para una sola moneda (Fila 1: el 1 y el 1) y siete resultados posibles para seis monedas (Fila 6: 1, 6, 15, 20, 15, 6 y 1). La suma de los números de la fila proporciona el número total de resultados (2 para una moneda, 4 para dos monedas, etc.). Cada número de la fila es una probabilidad. Por ejemplo, en la Fila 2, el primer número (1) es la probabilidad de obtener dos caras, el segundo número (2) es la de cara y cruz, y el tercer número (1) es la de dos cruces. Naturalmente estos números nos dan la frecuencia que hay que dividir por el número total de resultados posibles (en nuestro caso, cuatro) si queremos hallar la probabilidad propiamente dicha. Observa cómo el resultado de sumar los números de cada fila da siempre una potencia del dos (1, 2, 4, 8 y 16). Esto se explica porque para cualquier experimento, sólo existen dos sucesos posibles.



Ya puedes ir percatándote de lo útil que resulta este método cuando se trata de calcular las probabilidades resultantes de lanzar, por ejemplo, 30 monedas al aire, pero sería algo aburrido intentar resolverlo mediante la construcción de un triángulo de 30 filas, aparte del espacio que esto te iba a ocupar. Existe, en su lugar, un método gráfico que nos permite encarar tales casos, y aquí es donde puedes hacer intervenir a tu ordenador.



CURVAS DE DISTRIBUCION

Cuando existen muchos sucesos, y sus probabilidades no parecen tan evidentes, a menudo pueden obtenerse buenos resultados mediante el trazado de una curva de distribución. Ésta se traza mediante la frecuencia de los sucesos que se tengan tabulados: es la distribución de frecuencias. Como sucede con todo método gráfico, de un vistazo puede obtenerse la mayor parte de la información necesaria. Si, por ejemplo, vas a jugar al lanzamiento de una moneda 30 veces (que es lo mismo que lanzar 30 monedas de una sola vez), puedes visualizar el número de caras (o cruces) que salen en cada juego de 30 lanzamientos. Para ver el resultado, escribe la siguiente sección del programa, sin borrar la anterior sección:

```
170 REM Grafica de Picos
175 PLOT 0,0: DRAW 180,0
180 FOR x=4 TO 160 STEP 4
190 LET gm=0: GO SUB 610
200 FOR n=0 TO h: PLOT x,n*6:
    NEXT n
220 NEXT x
230 STOP
610 REM Lanzamiento
620 LET h=0: LET t=0
630 PRINT AT 4,22;"CARA:-";
    h;AT 6,22;"CRUZ:-";t
640 IF gm<>0 THEN PRINT AT
    0,0;"JUEGOS:-";gm;AT 21,
    3;"CARAS DE 30 TIRADAS:"
650 FOR s=1 TO 30
660 IF RND>=.5 THEN LET
    h=h+1: PRINT AT 5,31;
    "C";AT 4,29;h;" ": GO TO
    670
665 LET t=t+1: PRINT AT 5,31;
    "Z";AT 6,29;" "
670 NEXT s
680 RETURN
```



Ejecuta el programa, entrando ahora el 3 para seleccionar la tercera prueba. Verás un gráfico con una sucesión de puntos que ascienden hasta varios puntos elevados sobre la pantalla. Ésta es una de las muchas figuras posibles en este tipo de análisis. Los puntos elevados son el número de caras que se obtienen de 30 lanzamientos y se trazan a lo largo del eje Y, extendidas a lo largo del eje X. Observa que hay más cumbres altas que bajas. La razón de esto es que la probabilidad de obtener 15, o bien entre 12 y 17 caras es mucho más elevada que la de obtener un número menor o mayor de caras. Esto mismo es lo que puede verse en los números del triángulo de Pascal, con valores mayores justo para los números de enmedio.

La línea 180 establece un bucle para extender los puntos elevados a lo largo del eje X. La variable GM se inicializa a cero y señala el número de juegos de 30 lanzamientos (línea 190) y se llama a una rutina (líneas 610 a 680) donde se realiza cada juego de 30 lanzamientos. Esta rutina emplea los elementos de la segunda prueba, pero lanza la moneda «electrónica» 30 veces, en lugar de 100. Cuando ejecutes este programa verás que las letras H y T (cara y cruz) aparecen en la esquina superior derecha de la pantalla. Una vez realizados los 30 lanzamientos, el número de caras que se acumulan en la rutina se someten a una escala en la línea 200 y se trazan (PLOT) en la línea 210 como coordenadas de Y. Para sacar todo el jugo de este estudio, precisas ordenar la información de modo que obtengas una de las más conocidas curvas estadísticas: la distribución normal. Escribe estas pocas líneas y verás la curva de que te hablamos:

```
450 REM Distrib. normal
460 DIM 9(30)
470 PLOT 4,150: DRAW 0,
    -140: DRAW 245,0
480 GO SUB 560
485 IF INKEY$<>CHR$ 32
    THEN GO TO 485
```

```
560 REM Grafico
570 PLOT 4,10: FOR x=0 TO
    1200 STEP 20
580 DRAW 4,600*FNn(ABS((x-
    600)/140))+10-PEEK
    23678
590 NEXT x: RETURN
600 DEF FN n(x)=1/(PI*1.4142
    *2.718^((x^2)/2))
```

Ejecuta el programa y entra un 4 para ver una curva ideal del tipo distribución normal. La línea 460 da dimensión a una tabla, necesaria más adelante para guardar en ella la cuenta de caras obtenidas. La línea 470 dibuja dos ejes de coordenadas X-Y, y la línea 480 llama a una rutina para que dibuje la curva. Esta rutina emplea una función matemática (línea 580) para dibujarla, lo cual explica su forma ideal: une todos los puntos para obtener una curva bien alisada. La función está definida en la línea 600. No pulses todavía ninguna tecla, pues la rutina está incompleta y te dará error.

Muy rara vez puede obtenerse una curva perfecta al representar la información de los datos disponibles. Esto es de esperar, pues estás tratando probabilidades, no certezas. La probabilidad de un suceso (como, por ejemplo, la de tener lluvias torrenciales durante el período monzónico en la India) es elevada, pero han existido períodos en que lo que se ha dado es sequía en lugar del esperado diluvio. La siguiente prueba ilustra perfectamente este punto. Lo que hace es repetir el experimento un buen número de veces y representar gráficamente los resultados. Escribe la segunda parte de nuestra cuarta prueba:

```
490 FOR g=1 TO 200: LET
    gm=g
500 GO SUB 610
510 LET g(h)=g(h)+1
520 PLOT 8+8*h,10+4*g(h)
530 PRINT AT 21,25;h;" "
540 NEXT g
550 STOP
```

Ejecuta ahora la cuarta parte de nuevo. Una vez dibujada la curva ideal, pulsa la barra espaciadora o SPACE para iniciar el lanzamiento. Observa ahora la serie de puntos que va «creciendo» hasta llenar el espacio dentro de la curva. Cuando se completa la prueba, se habrán dibujado 200 puntos (PLOT de la línea 490). En algunos micros, y el Spectrum es uno de ellos, el tiempo que se invierte en este experimento es de varios minutos. Ésta es la razón por la que se ha escogido este número de 200 en lugar de otro más alto, como el 500 en la línea 490.

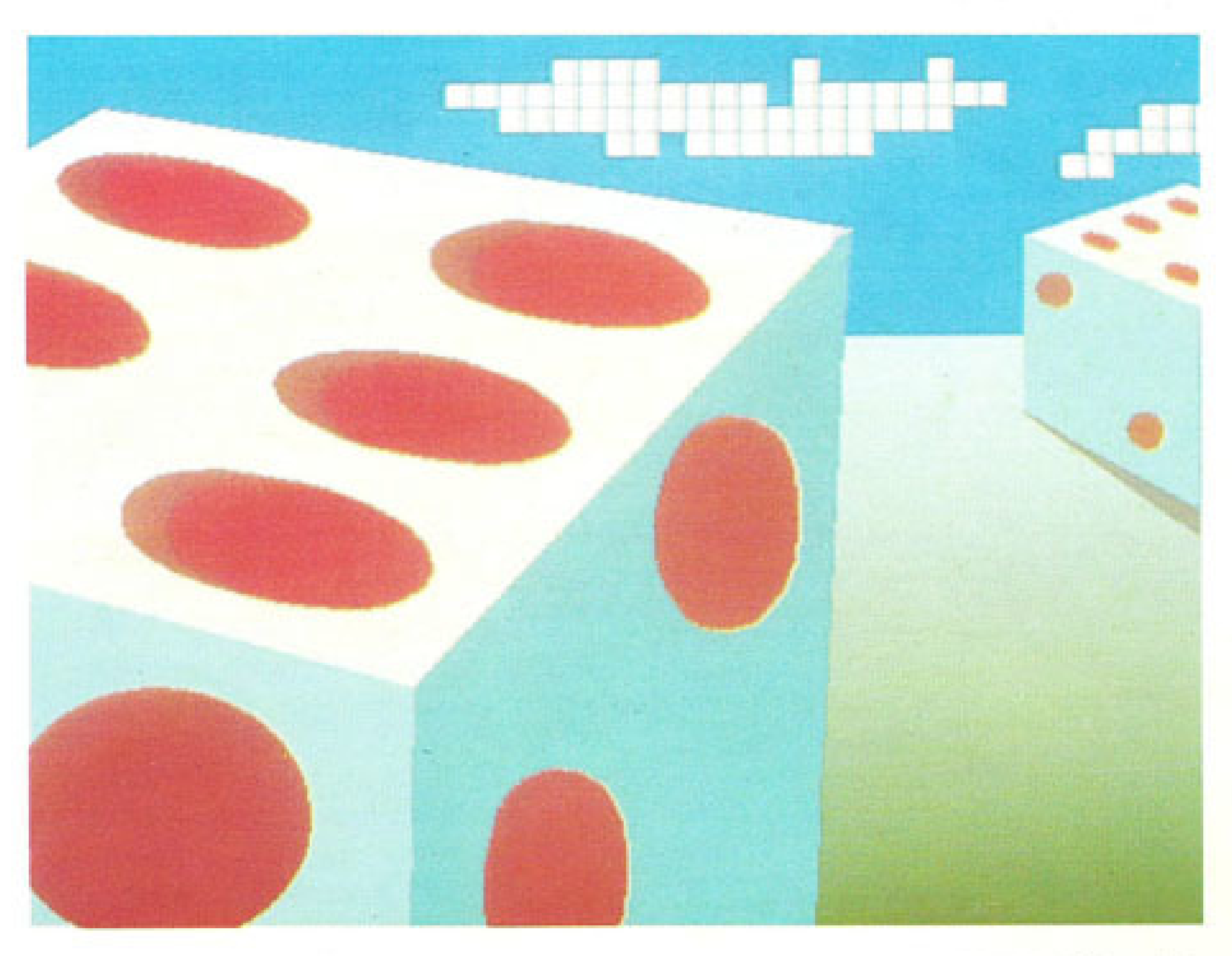
Esta parte del programa llama a la rutina (Línea 500) que lanza la moneda 300 veces de modo que, al igual que sucedía en la tercera prueba, los lanzamientos son iluminados en la parte superior derecha de la pantalla. Cada serie de 30 lanzamientos es un juego, y puede dar un número cualquiera de caras entre 0 y 30. El Spectrum no tiene el 0 en la primera variable de la tabla, como lo tienen otros ordenadores, por lo que hemos de prever el suceso raro de que no se obtenga ni una sola cara en 30 lanzamientos. Pero como este resultado es tan raro, a tenor de como se generan los números aleatorios, no lo hemos previsto, ni falta que hace. De hecho, son improbables números de caras o muy alejados de 30 o muy cercanos a este número total de lanzamientos: *puede* suceder, pero su probabilidad es mínima.

Mediante la tabla, la línea 510 guarda la cuenta de los resultados de cada juego. Por ejemplo, cada vez que el resultado de un juego es 11 caras, la casilla G(11) se incrementa en una unidad, y lo mismo pasa con la G(15) si el resultado fue de 15 caras. Al inicio todas las casillas están a 0.

Tras cada juego, la línea 520 somete a escala el valor de H (el número de caras en 30 lanzamientos) para obtener las coordenadas X e Y. La siguiente vez que sucede el mismo resultado se traza un punto en la misma posición de X, pero una unidad más adelante en el eje de Y. La línea 530 guarda la cuenta de H para cada juego, y también la cuenta del número de juegos efectuados.

EMPLEO DE LA CURVA

Ejecuta la cuarta prueba unas cuantas veces para ver cómo varía el perfil de los puntos dentro de la curva, y después haz lo mismo pero con valores finales de GM más pequeños (línea 490). Aun sin la curva ideal, pronto podrás imaginar una curva idealizada



a través de los puntos elevados. En la práctica, sin embargo, la inversa de este proceso imaginario es lo que resulta de un extraordinario valor: sabiendo el perfil de una curva, predecir los resultados de un experimento futuro.

LA MEDIA MATEMATICA

El valor de H en un punto central es de especial interés. Es la denominada media matemática, de los 31 valores H posibles a lo largo del eje X. En este caso, es 15. La media se identifica con el punto máximo de la curva. Es el valor con mayor probabilidad, pero no es útil por sí mismo. Se puede decir que 15 es lo más probable, pero también diremos que 14 y 16 son casi tan probables. Hay varios valores comunes en torno al máximo, y esto sí que es útil, saber su grado de dispersión. Por tanto la media se utiliza para especificar otro parámetro de vital importancia: la desviación típica, o la medida de la dispersión. La fórmula de la desviación típica es complicada, pero no sin una razón. Una vez calculado este parámetro, puedes asignar probabilidades a todos los puntos de la curva.

La desviación típica es una medida de la variación de los valores a ambos lados de la media. Por ejemplo, una sección de la curva con una desviación de 1,96 en cada lado de la media incluirá el 95 % de los resultados. Si amplías la variación típica a 2,58 la curva incluirá el 99 % de los resultados. Si empleas un software de estadísticas comerciales, la obtención de la desviación típica es de una extraordinaria facilidad.

UN CASO CON SEIS SUCESOS

Hay muchos ejemplos de experimentos que tienen más de dos sucesos posibles, cosa que no ocurriría en el lanzamiento sencillo de una moneda. En estos casos, determinar la probabilidad de un suceso no es tan fácil, ni se soluciona recurriendo a la respectiva fila del triángulo de Pascal. Por ejemplo, en el caso de un dado, al tirarlo puedes obtener seis posibles resultados. Si el dado no está cargado, los seis resultados tienen la misma probabilidad. ¿Y los posibles resultados de tirar dos dados? Se pueden enumerar mediante una tabla, pero siempre has de notar que cuantos más resultados posibles haya más complicada es la tarea de determinarlos.

He aquí una tabla de todos los resultados posibles al tirar un par de dados a la vez:

		valor del primer dado					
		1	2	3	4	5	6
	1	2	3	4	5	6	7
	2	3	4	5	6	7	8
valor	3	4	5	6	7	8	9
del	4	5	6	7	8	9	10
segundo	5	6	7	8	9	10	11
dado	6	7	8	9	10	11	12

Como puedes ver, hay 36 posibles resultados (seis filas por seis columnas) aunque sólo 11 son diferentes. Pero hay más cosas que se deducen de esta tabla. Sólo hay una posibilidad contra 36 de obtener la suma más baja o la suma más alta (2 y 12 sólo aparecen en la tabla una vez), mientras que hay seis posibilidades contra 36 (o sea 1/6) de obtener 7 de suma. Y hay también 6 posibilidades contra 36 de obtener un doble. Es lo que ofrece la diagonal que va desde el número 2 (doble de doses) hasta 12 (doble de seises).

Combinando el teorema binomial y esta tabla, puedes calcular las probabilidades de muchos experimentos. Un buen ejemplo de tirada múltiple de dados se encuentra en el juego del Monopoly. Si caes en la cárcel, tienes tres oportunidades para obtener un doble, de lo contrario has de pagar una fianza. Intuitivamente parecería que tienes un 50 % de probabilidades de salir de la cárcel (3 tiradas, con un 1/6 de probabilidad cada vez), pero no es así. En la tabla puedes ver que la probabilidad de que *no* obtengas un doble en cada tirada es de 30/36 (5/6). Con el teorema del binomio, puedes ver que la posibilidad de no obtener un doble las tres veces juntas es 5/6 elevado a 3, o sea, 125/216. Lo que da alrededor de un 58 % de posibilidad de fracaso, por tanto si tu situación financiera te lo permite, es preferible que no tientes a la mala suerte y pagues la fianza para librarte de tener que estar entre rejas.

LISTA DE PREMIADOS EN LA ENCUESTA DE JUNIO

Primer premio de 25.000 pts. en metálico:

ANA VALDEHITA VILLOTA. Madrid

Diez premios de un juego para cada concursante:

JOSE MANUEL FERNANDEZ ELVIRA. Alcalá de H. (Madrid)
ANTONIO DE LA NUEZ LA TORRE. Majadahonda (Madrid).
TORCUATO FERNANDEZ TEJADA
JAVIER VAZQUEZ ESTEBAN. Málaga
ALEJANDRO LOPEZ VINALS. H. de Llobregat (Barcelona)
JUAN ANTONIO GOMEZ MOHEDANO
FERNANDO GARCIA ABRAIDO TEJEDOR. Zaragoza
IGNACIO MIGUEL MONTILA. Carrión (Palencia)
FRANCISCO SANTOS JIMENEZ. Madrid
JUAN ANTONIO GOMEZ YELAMOS. Málaga

CONOS, CURVAS Y SECCIONES

- CORTAR UN CONO
- EL CIRCULO Y LA ELIPSE
- UNA PARABOLA Y UNA HIPERBOLA
- ROTACION DE CURVAS

La sencilla figura del cono es una de las formas geométricas más fascinantes de las matemáticas, y proporciona una entera gama de curvas. Con estos programas podrás investigar sus propiedades.

Las curvas han fascinado a los matemáticos desde tiempos inmemoriales, y cuanto más sencillas y elegantes, más importantes se consideraban. Los antiguos matemáticos griegos eran muy aficionados a hacer las matemáticas tan sencillas como se pudiera, y por ello, cuando se descubrió que se podía obtener toda una familia de curvas (conocidas como secciones cónicas) de un simple tajo practicado sobre un cono, les pareció que los conos debían de revestir un significado especial.

De hecho, la belleza de estas curvas está en que no son puras abstracciones matemáticas, sino que tropezamos con ellas en la vida diaria, y nos proporcionan una precisa descripción de los fenómenos físicos.

Naturalmente, en la naturaleza se encuentran otras curvas sencillas que no son secciones de un cono. La forma de una cuerda o de una cadena que cuelga entre dos puntos es una de ellas. Se llama catenaria, y, aunque parece una parábola, ambas curvas tienen sutiles diferencias, de tal modo que son descritas mediante ecuaciones muy diferentes. Pero las secciones cónicas son importantes, ya que sirven para señalar el modo como se mueven las cosas, y, por ende, se necesitan para cualquier programa realista.

Algunas curvas son también útiles como objetos sólidos tridimensionales. Los cortes de un cono son bidimensionales, pero pueden rotar sobre sus ejes y ofrecer formas tridimensionales. Así el círculo se convierte en una esfera, con infinidad de usos, y la parábola se transforma en un paraboloide, empleado en cosas tan diversas como los faros de un coche, el juego de espejos de un telescopio, hornos solares, etcétera.

Este artículo se divide en dos partes. La primera describe cada una de las curvas y cómo dibujarlas en la pantalla, mientras la segunda ilustra el uso de estas curvas en simulaciones como la trayectoria de un canjilón unido a una escalera (una elipse), o de una persona que cruza el río a nado (una parábola).

Verás también cómo se dibujan formas espectaculares con el solo empleo de la hipérbola y de la elipse.



CORTAR EL CONO

Las cuatro curvas que se obtienen cortando transversalmente un cono (el círculo, la elipse, la parábola y la hipérbola) son muy distintas entre sí. El primero que las estudió con detenimiento fue el griego Apolonio, allá por el año 200 a.C.

El punto inicial es el que formarían dos líneas que se cruzan como formando una X. Si se las hace rotar sobre un eje de simetría (véanse las ilustraciones) se obtienen dos conos que pueden cortarse de cuatro maneras.

Si se practica un corte en ángulo recto con el eje de simetría, la sección que se obtiene es un *círculo*.

El corte realizado formando un ángulo entre 90° y la mitad del ángulo que forman las líneas iniciales (denominado ángulo semivertical del cono), proporciona una sección denominada *elipse*.

El corte realizado formando un ángulo con el eje igual al ángulo semivertical nos da una *parábola*.

Un corte con un ángulo menor que el semivertical da una sección con dos partes denominada *hipérbola*. La hipérbola tiene dos partes porque el corte afecta a ambos conos, el superior y el inferior.

Observa estos dos casos especiales. El corte que incluye el eje, es decir, que corta los conos en dos mitades, de arriba abajo, proporciona dos líneas rectas (las que sirvieron para generar los conos). Se trata de un caso especial de la hipérbola. En segundo lugar, haciendo un corte formando un ángulo de 90° con el eje y por entre los dos conos, lo que se obtiene es un punto, que es un círculo con radio cero.

Las figuras que te adjuntamos te serán suficientes para comprender cómo se obtienen todas estas formas geométricas descritas. Si te animas, tú mismo puedes construirte tus conos y realizar los cortes en diferentes direcciones, con servirte tan sólo de una cuartilla de papel. Los dos conos son necesarios para obtener una parábola, ya que con un solo cono obtendrás sólo una parte del modelo.

Todas las curvas se generan mediante sencillas ecuaciones, algunas de las cuales tú quizás ya conozcas.

EL CIRCULO

La ecuación de un círculo está dada por

X = A*COS θ
Y = A*SIN θ

donde A es el radio, X, Y un punto de la circunferencia, y θ (la letra griega *zeta* o *theta*), el ángulo formado por una línea fija, que habitualmente es el eje X.

Este primer programa dibuja un círculo de radio A, y centro el de la pantalla:

```
10 CLS
15 LET a=70
25 LET x=a: LET y=0
30 PLOT 127+x,70+y
40 FOR t=0 TO 2*PI STEP .2
50 LET x=a*COS t: LET
   y=a*SIN t
60 DRAW x-PEEK 23677+127,y-
   PEEK 23678+70
70 NEXT t
```

LA ELIPSE

La ecuación de una elipse es muy similar a la del círculo. Para una elipse con el eje mayor igual a 2A y el eje menor igual a 2B, la posición de cualquier punto está dada por:

X = A*COS θ
Y = B*SIN θ

La forma más o menos achatada de la elipse se determina por A y B. Cambia estas líneas:

```
16 LET b=40
50 LET x=a*COS t: LET
   y=b*SIN t
```

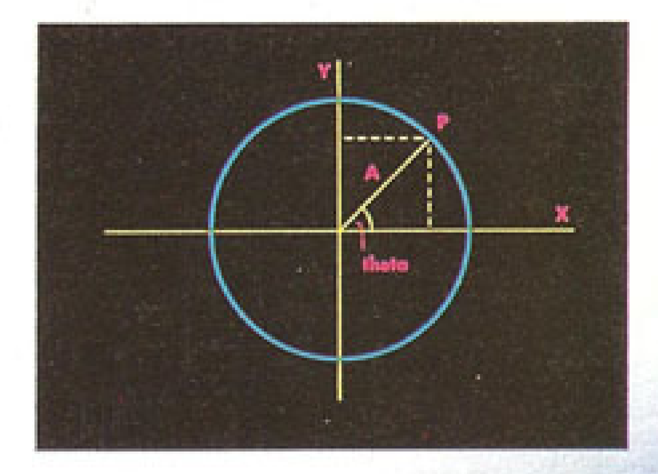
LA PARABOLA

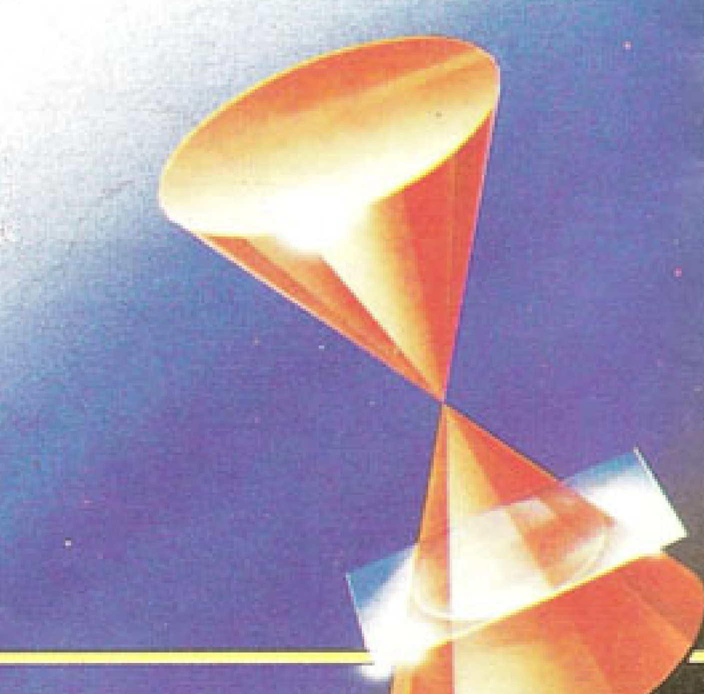
El tamaño de la parábola depende del valor de una variable T, y las ecuaciones son:

$X = T^2$
Y = 2*T

El valor T puede variar desde infinito hasta menos infinito, pero podemos hacer una sección de parábola bastante asequible entre T = 2 y T = −2. Estos valores deben ajustarse a escala en el programa mediante un factor M para que quepan en una pantalla de TV. Éste es el programa que dibuja la parábola:

```
10 CLS
15 LET m=20
25 LET x=4*m: LET y=-4*m
30 PLOT 127+x,80+y
40 FOR t=-2 TO 2 STEP .05
50 LET x=m*t*t: LET y=2*m*t
```





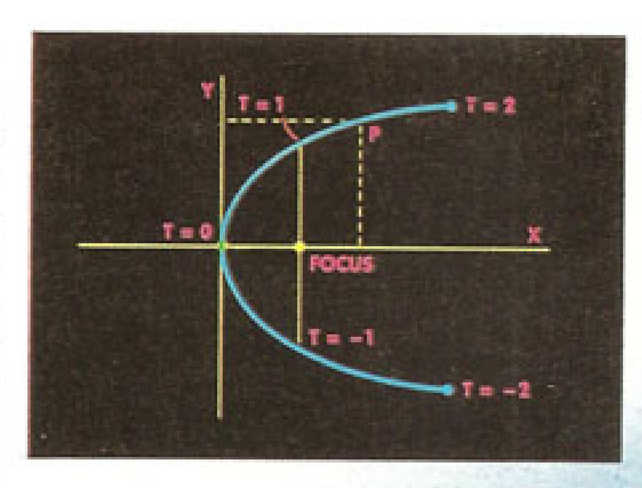
```
6Ø DRAW x-PEEK 23677+127,
   8Ø+y-PEEK 23678
7Ø NEXT t
```

LA HIPERBOLA

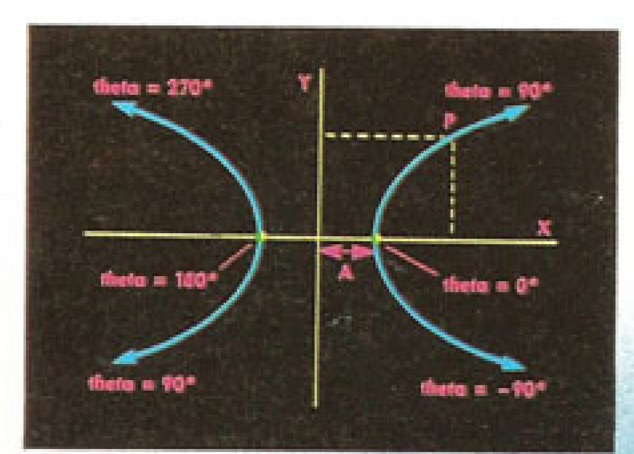
La hipérbola tiene por ecuación:

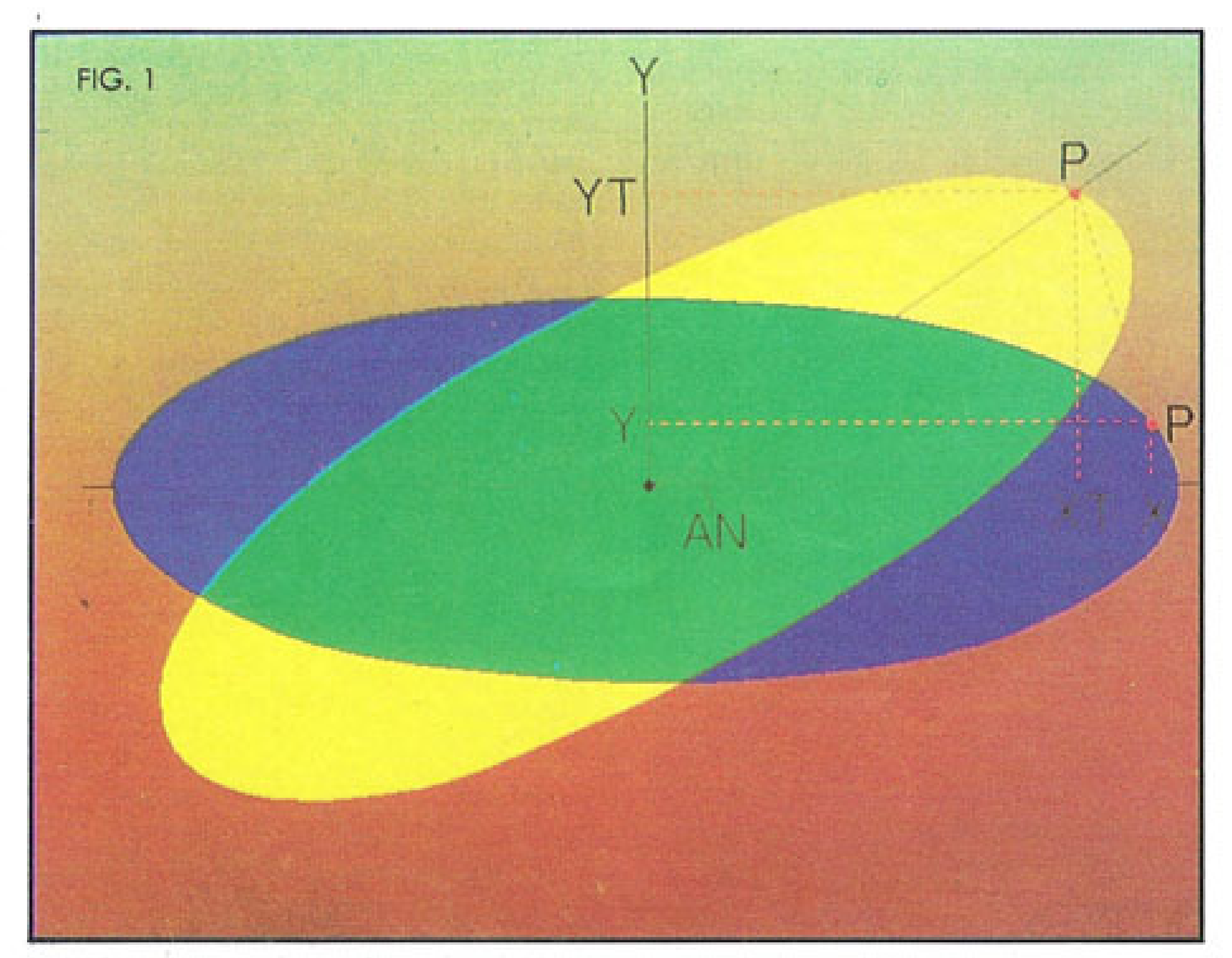
X = A/COS θ
Y = B*TAN θ

Una mitad de la hipérbola se obtiene cuando θ va de menos 90° a más 90°, y la otra mitad cuando va de más 90° a más 270°. Teóricamente es posible emplear sólo un bucle en el programa que lleve a θ desde −90°, 90° y 270°, donde se divide por cero, lo que el ordenador no sabe resolver. Incluso para valores de θ muy próximos a éstos, se obtienen valores enormemente altos. El programa que presentamos emplea, por ello, dos bucles. Y nuevamente, mediante el factor M, se puede dibujar a la deseada escala esta figura en la pantalla:




```
1Ø CLS
15 LET m=3Ø
25 LET x=m/COS -1: LET
   y=m*TAN -1
3Ø PLOT 127+x,75+y
4Ø FOR t=-1 TO 1 STEP .1
```






```
5Ø LET x=m/COS t: LET
   y=m*TAN t
6Ø DRAW 127+x-PEEK 23677,
   75+y-PEEK 23678
7Ø NEXT t
75 LET x=m/COS (PI-1): LET
   y=m*TAN (PI-1)
8Ø PLOT 127+x,75+y
9Ø FOR t=PI-1 TO PI+1 STEP .1
1ØØ LET x=m/COS t: LET
    y=m*TAN t
11Ø DRAW 127+x-PEEK 23677,
    75+y-PEEK 23678
12Ø NEXT t
```

HAGAMOS GIRAR LAS CURVAS

Los programas anteriores dibujaron las figuras de la manera más sencilla posible, con X de eje horizontal e Y de eje vertical. Pero esto no siempre es lo que conviene, pues puede que necesites dibujar las curvas formando un determinado ángulo. La fig. 1 muestra lo que sucede a un punto del borde de una elipse cuando se le hace girar con un ángulo de AN grados. El punto P se mueve de la posición X,Y a su nueva posición XT,YT y sus nuevas coordenadas son:

XT = X*COS AN − Y*SIN AN
YT = X*SIN AN + Y*COS AN

He aquí la rutina de la rotación para tu ordenador:

```
1ØØØ LET xt=x*COS (an*PI/18Ø)
     -y*SIN(an*PI/18Ø)
1Ø1Ø LET yt=x*SIN (an*PI/18Ø)
     +y*COS(an*PI/18Ø)
1Ø2Ø RETURN
```

En el programa que te dibuja la curva deberás introducir algunas modificaciones para que te sirva la rutina de la rotación. Se debe especificar (línea 17) el ángulo de rotación AN, la posición inicial ha de girar, y las líneas deben dibujarse según las nuevas coordenadas XT e YT en lugar de X e Y. Si lo deseas, puedes alterar la línea 17 para que te permita introducir (INPUT) el ángulo de giro.

He aquí los cambios que debes hacer en el programa que dibuja la elipse.

No olvides añadir la rutina de la rotación:

```
17 LET an=6Ø
28 GO SUB 1ØØØ
3Ø PLOT 127+xt,7Ø+yt
55 GO SUB 1ØØØ
6Ø DRAW xt-PEEK 23677+127,
   yt-PEEK 23678+7Ø
8Ø STOP
```

```
17 LET an=6Ø
28 GO SUB 1ØØØ
3Ø PLOT 127+xt,8Ø+yt
4Ø FOR t=-1.75 TO 1.75 STEP
   .Ø5
55 GO SUB 1ØØØ
6Ø DRAW 127+xt-PEEK 23677,
   8Ø+yt-PEEK 23678
8Ø STOP
```

```
17 LET AN=6Ø
28 GO SUB 1ØØØ
3Ø PLOT 127+xt,75+yt
55 GO SUB 1ØØØ
6Ø DRAW 127+xt-PEEK 23677,
   75+yt-PEEK 23678
76 GO SUB 1ØØØ
8Ø PLOT 127+xt,75+yt
1Ø5 GO SUB 1ØØØ
11Ø DRAW 127+xt−PEEK
    23677,75+yt−PEEK
    23678
13Ø STOP
```

APLICACIONES PRACTICAS

Todas estas curvas pueden tener ciertas aplicaciones prácticas.

El círculo tiene tantos usos que no es posible enumerarlos. La rueda es un ejemplo demasiado trivial de un círculo, y un cojinete de bolas es también otro uso obvio de la esfera. Las esferas, o sus aproximaciones, se presentan con frecuencia en la naturaleza. Los ejemplos que se pueden poner van desde las gotas de lluvia y los granos de guisante hasta los planetas. Pero las esferas pocas veces son perfectas debido al efecto de la gravedad, el viento u otras fuerzas. Un planeta que gira en torno a una estrella podría describir una circunferencia, pero lo más frecuente es que describa una elipse.

Una aplicación práctica de gran utilidad del círculo es la que permite determinar el menor coste de transporte de una mercancía que puede comprarse en uno de dos puestos de distribución. Por ejemplo, supongamos que tú deseas comprar un ordenador que se vende tanto en la casa A como

en la B y que distan entre sí 300 kilómetros. La casa A envía sus ordenadores a través de medios de transporte especiales y a un precio de 3 pts. por km, mientras que la casa B emplea su propia furgoneta, que viene a salir a 1,5 pts./km. Es inmediato marcar el área en el mapa de donde se puede obtener el ordenador deseado, al menor precio de transporte, sea de la casa A o de la B. La idea es marcar todos los puntos donde ambos costes son iguales, y unirlos por una línea. Lo que habrás hecho es marcar los puntos en que la distancia hasta B es el doble que la distancia hasta A.

Un punto estará en la línea entre A y B, a 100 km de A y 200 km de B (ya que 100×3 es igual que 200×1,5). Otro punto está en la misma línea a 300 km de A pero en la dirección opuesta a B (300×3 es igual que 600×1,5). Si tú unes todos estos puntos obtendrás un círculo de radio 200 km, conforme se muestra en la fig. 2. Si resides dentro del círculo es más barato comprarle a A, y si vives fuera de él es más barato comprarle a B.

La elipse tiene también sus usos prácticos. Si proyectas la sombra de una elipse sobre una superficie plana es posible mantener la elipse de tal manera inclinada que su sombra dé un círculo. Esta propiedad se utiliza en la válvula de los canales circulares, donde se utiliza una aleta elíptica para controlar el gas o el fluido que circula. La aleta se ajusta a la tubería justo cuando alcanza el ángulo adecuado y de esta forma cierra la tubería.

PROPIEDADES DE LA PARABOLA

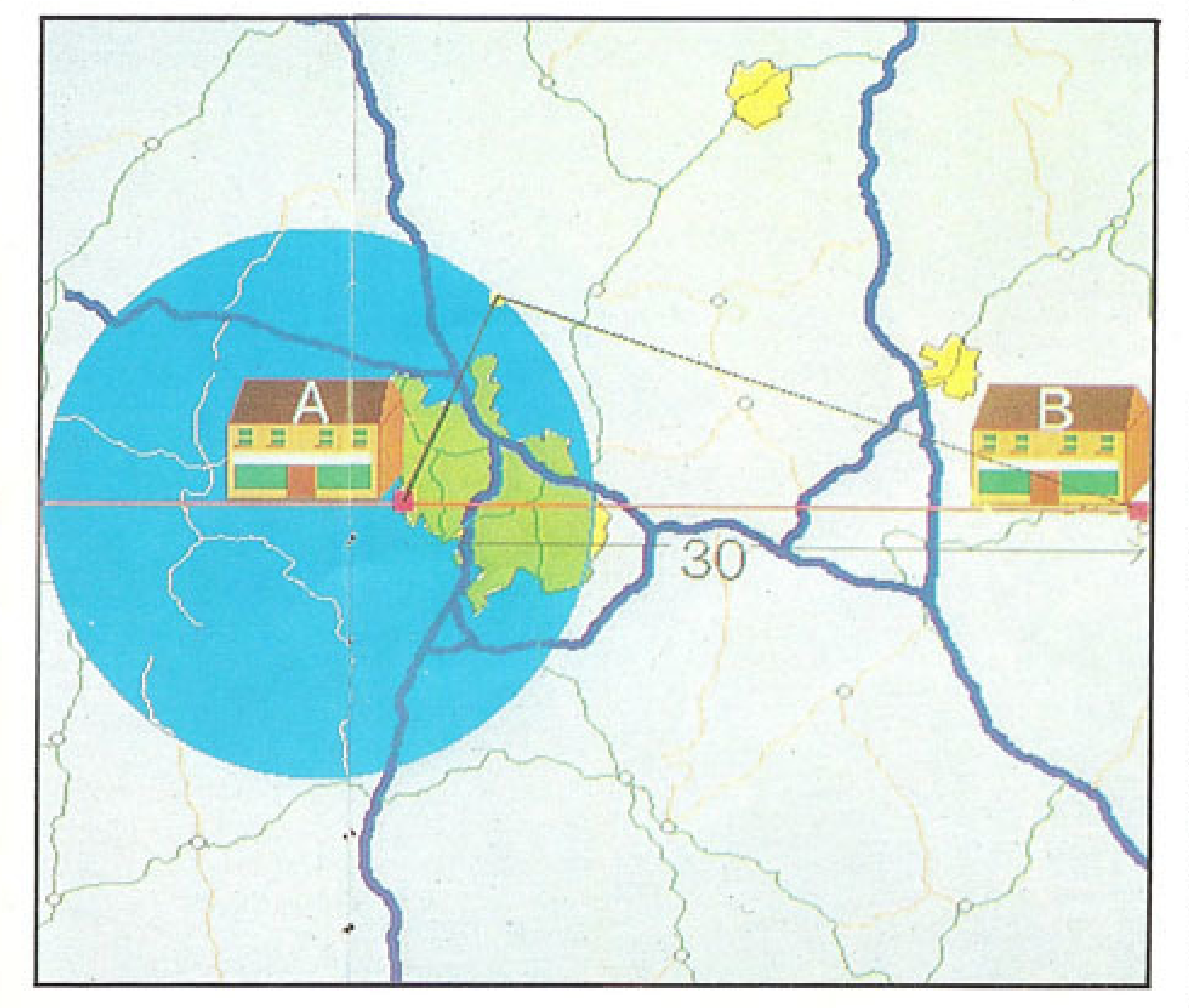
La parábola describe, como sabes, la trayectoria de un proyectil disparado al aire. Los cometas pueden viajar en forma parabólica alrededor del Sol.

Una propiedad muy útil de la parábola es que los rayos de luz, calor u otro cuerpo paralelo al eje se reflejan a través del cono. Esta propiedad es válida en ambos sentidos, por lo que una bombilla eléctrica colocada en el foco dará un haz de luz paralelo, como el que se usa en los faros de los automóviles. En la otra dirección, los rayos paralelos que proceden del sol se pueden concentrar en el foco para obtener altísimas temperaturas como ocurre en los hornos solares.

En la práctica, los reflectores empleados para tales propósitos son paraboloides tridimensionales. Otro uso de los paraboloides se halla en los discos que sirven de antena de radar o de radio, donde la antena se coloca en el foco y puede emplearse tanto para transmitir como para recibir señales.

Una importante característica de la hipérbola consiste en el hecho de estar formada de dos partes. Y un uso práctico de ello está en el sistema de radar de los barcos. El sistema se basa en dos estaciones de radar. Una estación transmite señales normalmente, y la otra sólo se limita a retransmitir las señales recibidas por la primera estación. Cualquier barco que se halle en las proximidades recibe ambas señales y anota el tiempo que media entre sus llegadas. Si continúa moviéndose de tal modo que conserve esta diferencia de tiempo constante, seguirá una trayectoria hiperbólica como la que mostramos en la fig. 3. Si el barco recibe también las señales de otras dos estaciones de radar y de nuevo toma nota de la diferencia de tiempo hará posible una segunda hipérbola y la intersección de ambas ofrecerá la posición del barco.

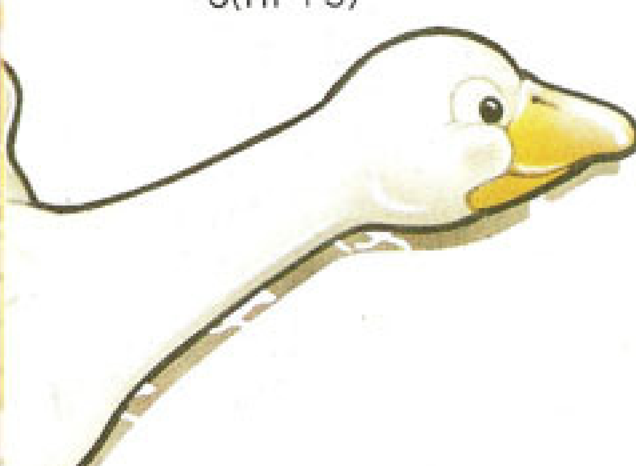
No puede haber confusión sobre la rama de la hipérbola en que se encuentra el barco puesto que puede detectarse la señal que primero llega.



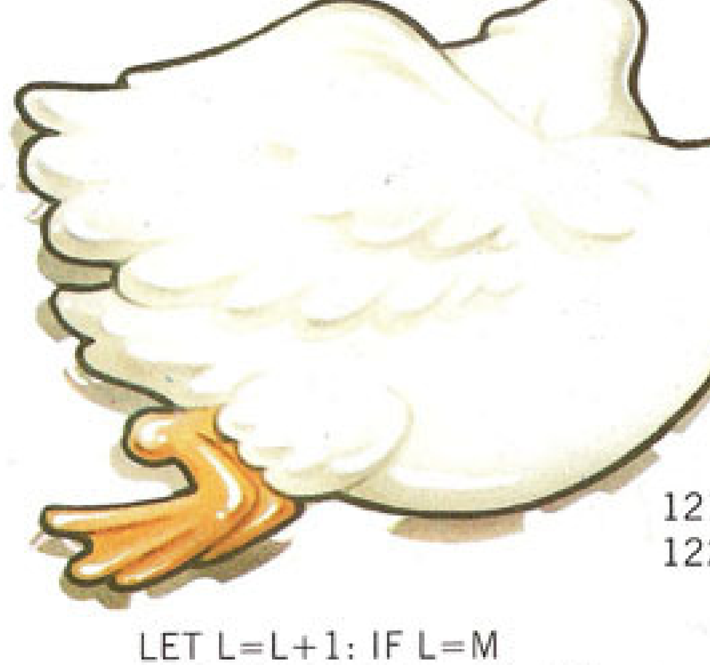
ADVERTENCIA FINAL

Los programas que te ofrecemos han sido pensados para una perfecta utilización de la pantalla de TV. Cuando los uses en tus propios programas, habrás de cambiar el factor M de incremento, para que las curvas se dibujen al tamaño que deseas; o debes tener también cuidado con la parábola que gira o la hipérbola, cerciorándote de que los cabos de las curvas entren en la pantalla. Para evitar que se salgan, altera los finales de los bucles FOR ... NEXT en la línea 40 del programa de la parábola y las líneas 40 y 90 del programa de la hipérbola. Para encontrar los límites exactos deberás recurrir al método de ensayo y error.

```
1Ø5Ø INPUT "MOVER EL ZORRO
     A ";B: IF B=-1 THEN GO
     SUB 271Ø: GO TO 1Ø3Ø
1Ø55 GO SUB 4ØØØ
1Ø6Ø FOR A=1 TO X(F): LET
     X=M(H,F): IF X=B THEN
     IF NOT (FN X(X)) THEN
     LET F=B: GO SUB 21Ø:
     GO TO 12ØØ
1Ø7Ø NEXT A: PRINT AT 21,Ø;
     "ESO NO ES LEGAL
          ": GO TO 1Ø5Ø
111Ø LET L=SF: LET M=SF:
     LET V(M)=E*M: IF M>4
     THEN DIM R(HF+3): DIM
     S(HF+3)
```



```
1112 GO SUB 112Ø: GO TO
     12ØØ
112Ø IF L=1 THEN GO TO 41Ø
1122 IF L<M-2 THEN GO SUB
     161Ø: IF V<>Ø THEN
     RETURN
113Ø LET L=L-1: LET
     V(L)=H*L: LET A(L)=X(F):
     LET F(L)=F
114Ø LET F=M(A(L),F(L))
115Ø IF FN X(F)=Ø THEN GO
     SUB 132Ø: IF V>V(L)
     THEN LET V(L)=V: LET
     P(L)=F: IF V>V(L+1)
     THEN LET F=F(L): LET
     L=L+1: RETURN
116Ø LET A(L)=A(L)-1: IF
     A(L)>Ø THEN GO TO 114Ø
117Ø LET V=V(L): LET F=F(L):
```



```
     LET L=L+1: IF L=M
     THEN LET F=P(M-1): GO
     SUB 21Ø: RETURN
1172 IF L<M-2 THEN GO SUB
     151Ø: RETURN
118Ø RETURN
```

MOVIENDO LAS OCAS

La rutina que va de la línea 1200 a la línea 1380 dirige el movimiento de las ocas. Hay rutinas para el caso en que el jugador controla las ocas —líneas 1220 a 1290— y para el caso en que es el ordenador quien controla las ocas —líneas 1320 a 1380.

```
5ØØ  LET V=Ø: FOR C=1 TO 4:
     LET G=G(C): FOR A=1 TO
     Z(G): LET X=M (A,G): IF
     X<>F THEN IF NOT FN X(X)
     THEN RETURN
5Ø2  NEXT A: NEXT C: LET V=1:
     RETURN
12ØØ GO SUB 31Ø: GO SUB 25Ø
12Ø2 IF F>28 THEN PRINT AT
     21,Ø;"EL ZORRO HA
     GANADO": GO TO 141Ø
12Ø4 GO SUB 5ØØ: IF V THEN
     PRINT AT 21,Ø;"EL
     ZORRO HA GANADO
          ": GO TO 141Ø
121Ø IF PG THEN GO TO 131Ø
122Ø INPUT "QUE OCA
     MUEVES?";G: IF G=-1
     THEN GO SUB 271Ø: GO
     TO 12Ø2
1225 GO SUB 4ØØØ
123Ø LET C=FN Z(G): IF C=Ø
     THEN PRINT AT 21,Ø;
     "NINGUNA OCA EN";G;"":
     GO TO 122Ø
124Ø INPUT "A DONDE";I: IF
     I=-1 THEN GO SUB
     271Ø: GO TO 12Ø2
125Ø IF FN X(I) THEN PRINT AT
     21,Ø;"NO PONER OTRA
     OCA                ": GO
     TO 122Ø
126Ø IF I=F THEN PRINT AT
     21,Ø;"NO PONER ZORRO
               ": GO TO 122Ø
127Ø FOR A=1 TO Z(G): IF M(A,
     G)=I THEN LET G(C)=I:
     GO TO 1Ø1Ø
128Ø NEXT A: PRINT AT 21,Ø;
     "ESO NO ES LEGAL
          ": GO TO 122Ø
131Ø LET L=SG: LET M=SG:
     LET V(M)=H*M: IF M>4
     THEN DIM R(HF+3): DIM
     S(HF+3)
1312 GO SUB 132Ø: GO TO
     1Ø2Ø
132Ø IF L=1 THEN GO TO 51Ø
1322 IF L<M-2 THEN GO SUB
```



```
      1610: IF V<>0 THEN
      RETURN
1324 LET L=L-1: LET
      V(L)=E*L: LET C=1
1330 LET C(L)=C: LET
      F(L)=G(C): LET A(L)=1: IF
      A(L)>Z(G(C)) THEN GO TO
      1362
1340 LET B=M(A(L),F(L)): LET
      X=FN X(B): LET G(C)=B:
      IF X OR B=F THEN GO TO
      1360
1350 GO SUB 1120: LET
      C=C(L): IF V<V(L) THEN
      LET V(L)=V: LET
      P(L)=G(C)+C*32
1355 IF V<V(L) THEN LET
      G(C)=F(L): LET L=L+1:
      RETURN
1360 LET A(L)=A(L)+1: IF
      A(L)<=Z(F(L)) THEN GO
      TO 1340
1362 LET G(C)=F(L): LET
      C=C+1: IF C<5 THEN GO
      TO 1330
1370 LET V=V(L): LET L=L+1:
      IF L=M THEN LET C=INT
      (P(L-1)/32): LET
      G(C)=P(L-1)-C*32: GO
      SUB 210: RETURN
1372 IF L<M-2 THEN GO SUB
      1510: RETURN
1380 RETURN
```

LOS MEJORES MOVIMIENTOS

Estas rutinas están relacionadas con el «un movimiento», en otras palabras, en esta fase, el ordenador sólo programa un solo movimiento cuando está buscando el mejor movimiento. Las líneas 410 a 420 repasan todos los posibles movimientos que puede realizar el zorro, utilizando la dirección M de trazado. Coloca la subrutina comenzando en la línea 2110 (ver anterior). La subrutina devuelve un valor de P, configuración posterior al mejor movimiento, y V, evaluación posterior al mejor movimiento.

Las líneas 510 a 530 son una rutina similar, pero en esta ocasión, el ordenador calcula cuál es el mejor movimiento para las ocas. P y V están fijadas de igual modo que en la rutina previa.

En ambos casos, GOSUB 210 recoge el mejor movimiento de entre todos los evaluados.

```
410 LET V=H: FOR A=X(F) TO 1
     STEP -1: LET X=M(A,F): IF
     FN X(X) THEN NEXT A: LET
     L=1: RETURN
420 LET B=F: LET F=X: GO
     SUB 210: LET V=P: LET
     F=B: LET L=1: RETURN
510 LET V=E: FOR C=1 TO 4:
     LET G=G(C): IF -B(G)>V
     THEN GO TO 530
520 FOR A=1 TO Z(G): LET
     X=M(A,G): IF FN X(X) OR
     (X=F) THEN NEXT A: GO TO
     530
528 LET V=B(X)-B(G):
     LET D=C: LET B=X
530 NEXT C: LET G=G(D): LET
     G(D)=B: GO SUB 210: LET
     V=P: LET G(D)=G: RETURN
```

LA TABLA HASHCODE

En los niveles superiores de juego preferirás aplicar el algoritmo alfabeta. En efecto, ya has utilizado previamente el algoritmo como parte de las rutinas de los movimientos del zorro y del de las ocas. Antes de aplicar el algoritmo, comprueba si es conveniente utilizar el algoritmo (¿el nivel de juego elegido es suficientemente alto como para garantizar su utilización?)

La rutina que va de la línea 1110 a la línea 1150 controla al zorro, mientras que la rutina que va de la línea 1310 a la línea 1350 controla las ocas.

El algoritmo se aplica en el último test IF al final de las líneas 1150 y

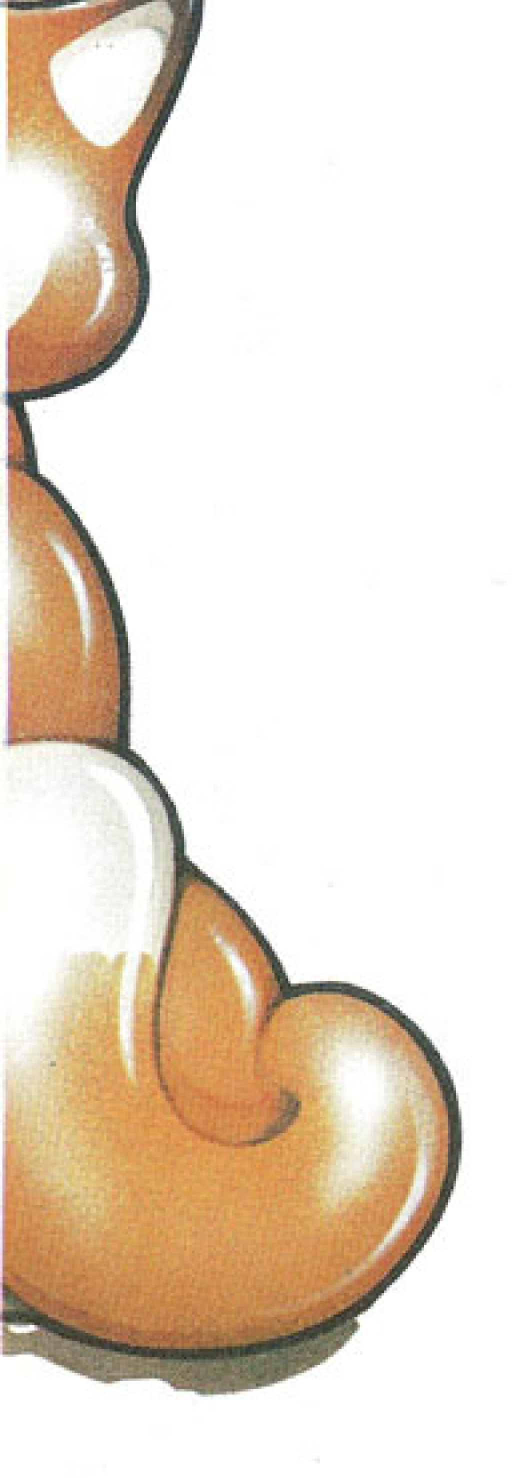


1350, después de que V(M) ha sido colocado al nivel apropiado en las líneas 1110 y 1310.



El algoritmo alfa-beta se utiliza conjuntamente con una técnica conocida como *hashcoding* con el fin de confeccionar y utilizar una tabla de los movimientos ya considerados. *Hashcoding* permite al ordenador comprobar rápidamente si el movimiento ya ha sido considerado. Cuanto más larga sea la tabla de *hashcoding* dentro del programa, más rápidamente se debería poder ejecutar el programa.

feccionar y utilizar una tabla de los movimientos ya considerados. Haschcoding permite al ordenador comprobar rápidamente si el movimiento ya ha sido considerado. Cuanto más larga sea la tabla de *haschcoding* dentro del programa, más rápidamente se debería poder ejecutar el programa.



La tabla se inicializa en las líneas 2500, 2750 y 2800. Hay *mejores valores* teóricos para las dimensiones de las direcciones que sostienen las tablas (línea 2500). Las direcciones han sido dimensionadas lo más largas posibles, indicando la memoria RAM disponible en cada ordenador.

La tabla es puesta a cero en las líneas 1110 y 1310, los contenidos se verifican en las líneas 1122 y 1322, y se colocan en las líneas 1172 y 1372.

La subrutina de verificación se inicia en la línea 1610 y la subrutina de ajuste en la línea 1510.

```
1510 GO SUB 210: LET C=P
1520 LET C=C-INT ((C/HF+C)
     -C)*HF: IF C<0 OR C>
     =HF THEN GO TO 1520
1550 FOR A=C+1 TO C+4: IF
     R(A)<>0 AND R(A)<>P
     THEN NEXT A: RETURN
1560 LET R(A)=P: LET S(A)=V:
     RETURN
1610 GO SUB 210: LET C=P
1620 LET C=C-INT ((C/HF+C)
     -C)*HF: IF C<0 OR
     C>=HF THEN GO TO
     1620
1650 FOR A=C+1 TO C+4: IF
     R(A)<>0 AND R(A)<>P
     THEN NEXT A: LET V=0:
     RETURN
1660 LET V=S(A)*(R(A)=P):
     RETURN
```



FREDDY Y LA ARAÑA DE MARTE (I)

- LA TRAMA
- COMO SE PROGRAMA EL JUEGO
- OBTENCION DE LOS GRAFICOS
- FREDDY, FLECHAS, GLOBOS Y LA ARAÑA

Abandona a su destino a Freddy y a la voraz araña de Marte mientras te ocupas de reconstruir uno de los juegos electrónicos más típicos. En el siguiente artículo tendrás el programa completo.

Freddy y la araña de Marte, o *La pesadilla del limpiador de cristales*, es un juego electrónico que, para ofrecértelo completo, nos va a ocupar dos artículos, pero a cambio vas a conocer un clásico de los juegos electrónicos.

En este primer artículo se establecen los gráficos y se inicializa el programa. En el siguiente, obtendrás el programa entero con sólo coser las rutinas entre sí.

EL JUEGO

El punto de partida para diseñar un juego es, claro está, una idea: una trama o un argumento pueden servir de base para construir el juego.



En el que vamos a examinar, Freddy es un limpiacristales que tiene pánico de las arañas. Ni las consultas al médico ni a innumerables especialistas han logrado que supere su fobia. Tan grave es ésta, que ahora le da por soñar pesadillas sobre una araña marciana, enorme, horrible y voraz, o por tener sueños con globos y lances con el tiro al arco, su deporte favorito.

A menudo se ha sorprendido sudando a chorros, despierto después de haber soñado que estaba acorralado, subido a su escalera y armado no de su habitual cubo de la limpieza sino de un arco y una aljaba repleta de flechas. Toda su ansia se había concentrado en hacer explotar esos globos temibles que, si logran llegar hasta la jaula de la araña que pende sobre su cabeza, está perdido, pues abrirán las puertas y liberarán al no menos temido arácnido.

O te afanas en ayudar al pobre Freddy o su final es tan seguro como apetitoso resultará a la araña el desayuno que imaginas.

Hasta aquí el argumento o la trama. En el marcador se anotarán tantos puntos cuantos globos se hayan conseguido estallar. Pero las pesadillas no suelen acabar tan fácilmente: la tortura del pobre limpiador de ventanas aumenta de grado a medida que los globos pasan delante de él cada vez más veloces. Con que falle en la destrucción de tres de éstos, la araña campará por sus respetos.

COMO SE ESCRIBE EL JUEGO

Son cuatro los objetos que poseen movimiento en la pantalla: la araña, que se mueve tanto horizontal como verticalmente; el globo, que surca el espacio y que sólo se mueve verticalmente; Freddy, que también se mueve en dirección vertical únicamente, y la flecha que éste dispara. La flecha se mueve en general horizontalmente, pero cuando a Freddy le da por subir o bajar por la escalera entonces la flecha habrá de desplazarse verticalmente junto con nuestro personaje. Sin embargo, quienes más nos deben preocupar son la araña y el globo de turno. Son muchas las variables que se asocian a estos dos objetos. La técnica que mejor se adapta a este caso es almacenar las variables en una tabla unidimensional, empleando una constante para referirnos a una casilla en particular.

En un segundo paso habrás de plantearte cómo visualizarás estas figuras en la pantalla. Lo que equivale a determinar cuántos dibujos diferentes se van a necesitar a lo largo del programa. Todos los grandes juegos para ordenadores incorporan dibujos extremadamente cuidados y de gran colorido, lo cual quiere decir que este segundo paso no has de tomarlo a la ligera. Para bajar a un buen nivel de detalle, construiremos la figura de Freddy con una ordenación tres por dos de caracteres UDG. Para visuali-



zar la araña utilizaremos figuras dos por dos (dispondremos de dos figuras para que la animación resulte más realista). Por último, están la forma del globo y el dibujo de un globo que explota. Con dos caracteres describirás la flecha y con otros dos, la escalera. Esto da un total de 26 caracteres.

En esta primera parte, tienes establecidos los gráficos e inicializado el juego.



INICIALIZACION DE LOS GRAFICOS

```
1000 DIM b(6): DIM s(7)
1010 LET xpos=1: LET ypos=2:
     LET colour=3: LET
     points=4: LET count=5:
     LET maxcount=6
1020 LET xinc=3: LET yinc=4:
     LET picture=7
1030 LET dest=65288
1040 FOR i=0 TO 26*8-1:
     READ j: POKE dest+i,j:
     NEXT i
1050 DATA 15,63,127,255,
     255,255,255,27
1060 DATA 240,252,254,255,
     255,255,255,254
1070 DATA 127,63,63,31,15,
     7,3,6
1080 DATA 254,252,252,248,
     240,224,192,96
1090 DATA 32,96,255,255,96,
     32,0,0
1100 DATA 5,10,252,252,10,
     5,0,0
1110 DATA 1,64,17,40,16,0,0,
     161
1120 DATA 128,2,136,20,8,0,
     0,133
1130 DATA 161,0,0,16,40,17,
     64,1
1140 DATA 133,0,0,8,20,136,
     2,128
1150 DATA 48,48,48,48,111,
     111,48,48
1160 DATA 12,12,12,12,246,
     246,12,12
1170 DATA 7,31,49,57,127,
     112,237,255
1180 DATA 224,248,140,204,
     254,14,183,255
1190 DATA 127,59,51,99,115,
     35,6,12
1200 DATA 254,108,102,51,
     49,25,24,48
1210 DATA 7,31,49,51,127,
     112,235,255
1220 DATA 224,248,140,156,
     254,14,215,255
1230 DATA 127,51,50,27,25,
     50,112,224
1240 DATA 254,204,108,102,
     54,22,3,6
1250 DATA 15,31,19,55,55,
     63,63,15
1260 DATA 240,248,248,252,
     252,252,252,240
1270 DATA 251,219,139,219,
     219,251,247,239
1280 DATA 252,254,254,254,
     254,254,254,252
1290 DATA 95,127,31,31,31,
     63,127,127
1300 DATA 188,188,188,188,
     188,124,252,248
1310 LET hiscore=0
1320 RETURN
```



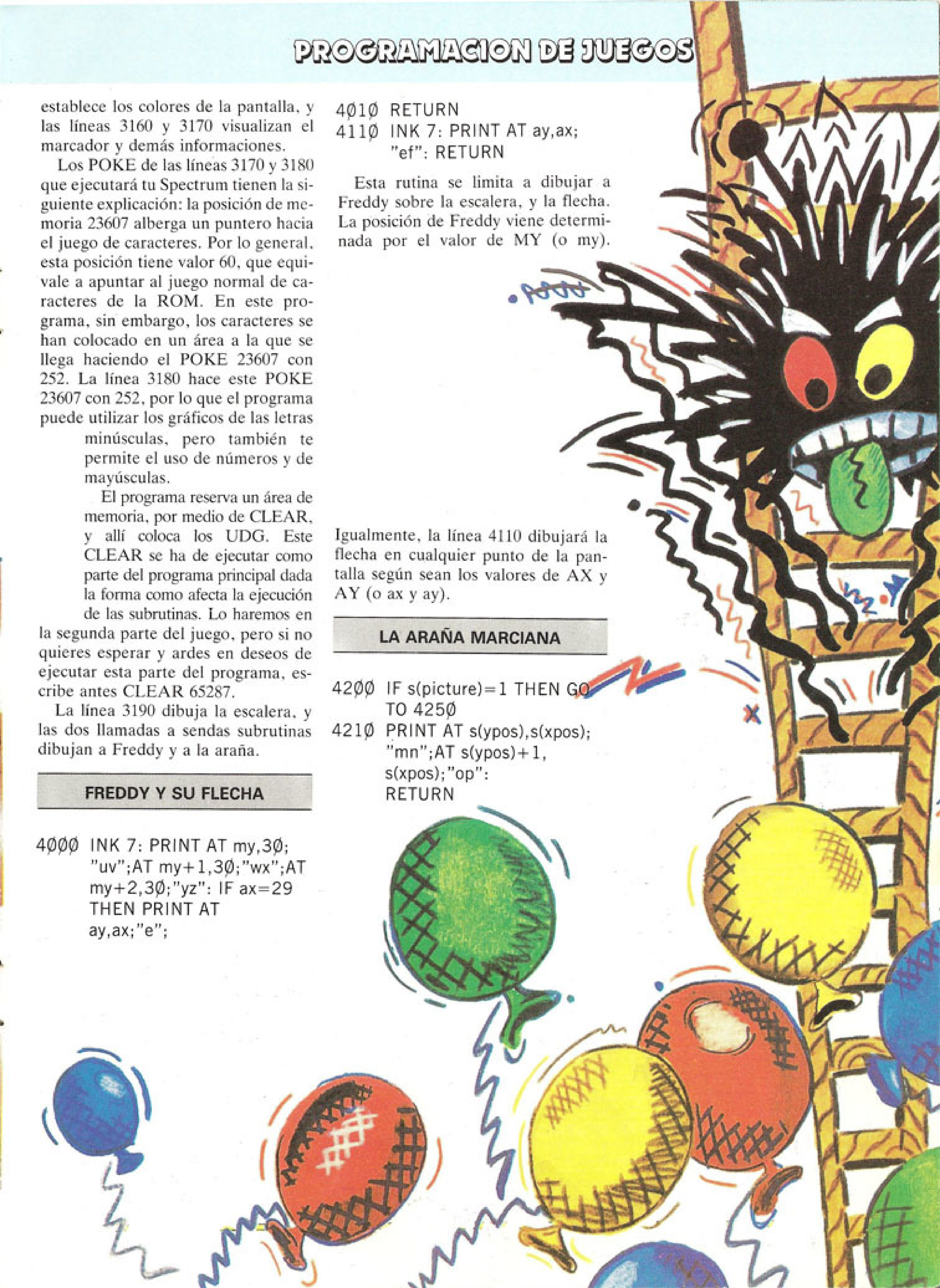
Este fragmento del programa almacena los DATA de los UDG correspondientes al globo y a la araña en las tablas B (o b) y S (o s): tablas que se DIMensionan en la línea 1000.

Las líneas 1010 a 1020 definen los valores iniciales de los punteros que apuntan a las tablas, antes de establecer los UDG. Por último, la línea 1310 coloca el marcador a cero. Se coloca esta línea aquí dado que sólo ha de ejecutarse una sola vez durante todo el programa.

INICIALIZACION DEL JUEGO

```
3000 LET score=0: LET level=1
3010 LET my=15
3020 LET bl=15+5*level: LET
     ax=29: LET ay=16: LET
     dead=0: LET props=3
3090 GO SUB 5000
3150 PAPER 0: BORDER 0: CLS
3160 FOR x=0 TO 28: PRINT AT
     3,x; INK 0; PAPER 6;" ";
     AT 0,x;" ": NEXT x: GO
     SUB 6000
3170 POKE 23607,60: PRINT
     AT 0,0; INK 0; PAPER 6;
     "L=";level;" B=";bl;" ";
     "SC=";score;AT 0,20;
     HI=";hiscore
3180 POKE 23607,252
3190 FOR y=5 TO 21: PRINT AT
     y,30; INK 6;"kl": NEXT y
3200 POKE 23607,252
3210 GO SUB 4000
3220 GO SUB 4200
3240 RETURN
```

Las líneas 3000, 3010 y 3020 colocan el marcador a cero y el nivel de dificultad a uno, al tiempo que inicializan una serie de variables. La línea 3150



establece los colores de la pantalla, y las líneas 3160 y 3170 visualizan el marcador y demás informaciones.

Los POKE de las líneas 3170 y 3180 que ejecutará tu Spectrum tienen la siguiente explicación: la posición de memoria 23607 alberga un puntero hacia el juego de caracteres. Por lo general, esta posición tiene valor 60, que equivale a apuntar al juego normal de caracteres de la ROM. En este programa, sin embargo, los caracteres se han colocado en un área a la que se llega haciendo el POKE 23607 con 252. La línea 3180 hace este POKE 23607 con 252, por lo que el programa puede utilizar los gráficos de las letras minúsculas, pero también te permite el uso de números y de mayúsculas.

El programa reserva un área de memoria, por medio de CLEAR, y allí coloca los UDG. Este CLEAR se ha de ejecutar como parte del programa principal dada la forma como afecta la ejecución de las subrutinas. Lo haremos en la segunda parte del juego, pero si no quieres esperar y ardes en deseos de ejecutar esta parte del programa, escribe antes CLEAR 65287.

La línea 3190 dibuja la escalera, y las dos llamadas a sendas subrutinas dibujan a Freddy y a la araña.

FREDDY Y SU FLECHA

```
4ØØØ INK 7: PRINT AT my,3Ø;
     "uv";AT my+1,3Ø;"wx";AT
     my+2,3Ø;"yz": IF ax=29
     THEN PRINT AT
     ay,ax;"e";
4Ø1Ø RETURN
411Ø INK 7: PRINT AT ay,ax;
     "ef": RETURN
```

Esta rutina se limita a dibujar a Freddy sobre la escalera, y la flecha. La posición de Freddy viene determinada por el valor de MY (o my).

Igualmente, la línea 4110 dibujará la flecha en cualquier punto de la pantalla según sean los valores de AX y AY (o ax y ay).

LA ARAÑA MARCIANA

```
42ØØ IF s(picture)=1 THEN GO
     TO 425Ø
421Ø PRINT AT s(ypos),s(xpos);
     "mn";AT s(ypos)+1,
     s(xpos);"op":
     RETURN
```

```
425Ø PRINT AT s(ypos),s(xpos);
     "qr";AT s(ypos)+
     1,s(xpos);"st":
     RETURN
```

Esta rutina se parece en todo a las restantes rutinas de impresión, pero necesita una atención especial porque juega con dos figuras. Ambas son impresas en el mismo lugar de la pantalla, una después de la otra, lo que hace que las patas del horrible bicho parezcan moverse.

UN GLOBO SE ELEVA

```
43ØØ PRINT AT b(ypos),b(xpos);
     BRIGHT 1; INK b(colour);
     "ab";AT b(ypos)+1,
     b(xpos);"cd": RETURN
5ØØØ LET range=INT (RND*6)
5Ø1Ø LET
     b(xpos)=(4*range)+INT
     (RND*4)
5Ø2Ø LET b(ypos)=2Ø
5Ø3Ø LET b(maxcount)=5-level
5Ø4Ø LET b(count)=1
5Ø5Ø LET b(colour)=INT
     (RND*5)+3
5Ø6Ø LET b(points)=1Ø-range
5Ø7Ø RETURN
```

La línea 4300 dibuja el globo en la pantalla, sin más.

Las restantes líneas preparan un nuevo globo en cuanto haya estallado el precedente o haya alcanzado la jaula de la araña. Los globos despuntan por la parte inferior de la pantalla en uno de los seis lugares previstos, elevándose acto seguido verticalmente. MAXCOUNT (o maxcount) determina la frecuencia de movimientos del globo, o sea su velocidad, y depende del nivel de dificultad elegido para el juego. En tu Spectrum se elige el color del globo y después se establece éste según el grado de aproximación a la escalera de Freddy que se escoja.



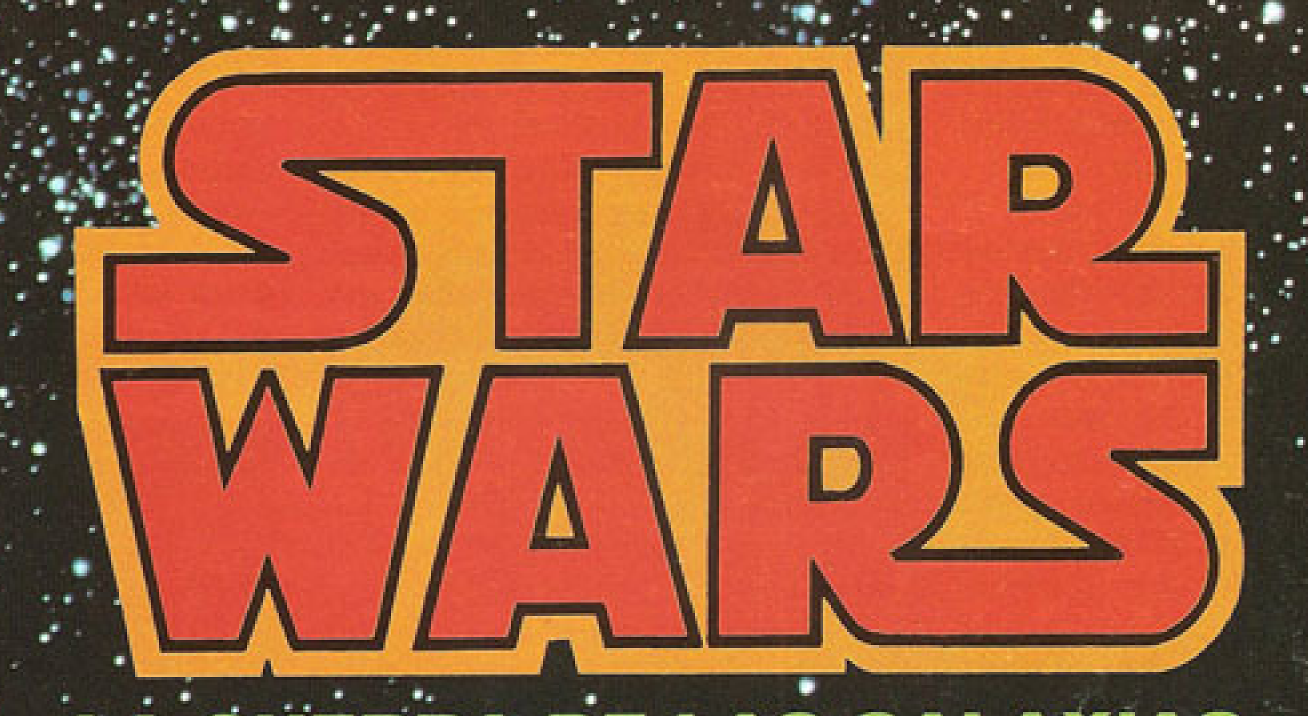
LAS PUERTAS

```
6ØØØ IF level<>1 THEN POKE
     236Ø7,6Ø: PRINT AT
     s(ypos),s(xpos);" ";AT
     s(ypos)+1,s(xpos);" ":
     POKE 236Ø7,252
6Ø1Ø FOR x=1Ø TO
     3Ø STEP 9
6Ø2Ø PRINT INK 6;AT 1,x;""
6Ø3Ø NEXT x
6Ø4Ø LET s=(xpos)=1: LET
     s(ypos)=1: LET s(xinc)=1:
     LET s(yinc)=Ø: LET
     s(count)=4: LET
     s(maxcount)=4: LET
     s(picture)=1
6Ø5Ø RETURN
```

Para retener prisionera a la araña se han dibujado tres puertas. Notarás que para tu Spectrum el carácter ? significa de hecho una explosión, y se obtiene, en el modo gráficos, mediante la tecla 8.



¡LA MAS GRANDE AVENTURA ESPACIAL DE TODOS LOS TIEMPOS!



forum

LA GUERRA DE LAS GALAXIAS

STAR WARS

JUVENIL

CADA MES EN TU QUIOSCO

¡DOS COMIC-BOOKS USA AL PRECIO DE UNO!

¡EL JUEGO MORTAL DE DARTH VADER!

Nº 7

140 PTAS

COMICS forum

ANIMACION MEDIANTE GRAFICOS PAGINADOS

La utilización de técnicas simples de gráficos paginados ofrece una real comprensión del mundo de la animación por ordenador y obtiene unos resultados interesantes en tu micro.

Todos los tipos de animación se basan en un fenómeno de percepción conocido como persistencia de la visión. En efecto, esto significa que una imagen que vemos se «conserva» en la memoria durante un instante apreciable, incluso aunque la visión se mueva hacia cualquier otro punto. Si se muestra rápidamente una secuencia de imágenes estáticas, el cerebro no puede retener los cambios de imagen que ocurren a una frecuencia superior a doce veces por segundo, aproximadamente. Como resultado, deja de separar las imágenes y, confundido, cree ver movimiento.

El proceso se demuestra con mucha sencillez con el «cambio de página», en el cual los dibujos de cada una de las páginas de un libro pueden ponerse en movimiento a medida que se pasan las páginas a una cierta velocidad. Una demostración más sofisticada de lo mismo se encuentra en el proceso de animación por cuadros.

Este tipo de animación consiste en dibujar una figura sobre una pieza de plástico, denominada *celda*, fotografiando a continuación un par de cuadros de película utilizando una cámara de cine convencional montada sobre una montura situada sobre dicha película. La celda se sustituye a continuación por otra que muestra una versión ligeramente alterada de la misma figura y se repite todo el proceso anterior. Como puedes imaginar, este tipo de animación requiere un increíble número de imágenes que deben dibujarse concienzudamente a mano, ya que se necesitan alrededor de unos veinticinco cuadros por cada segundo de película terminada.

GRAFICOS POR ORDENADOR

Por tanto, ¿por qué no utilizar los ordenadores para acelerar el proceso? Incluso el relativamente humilde micro doméstico puede generar buenas imágenes, mientras que los ordenadores especiales son capaces de obtener imágenes asombrosamente elaboradas.

El problema es que obtener una visualización de la calidad que da por supuesta hoy en día el público cinematográfico requiere una variedad fantásticamente extensa de hardware. Una película de ciencia ficción, *El guerrero del espacio (Starfighter)*, confió en un equipo informático para conseguir 27 minutos de sorprendentes imágenes. Pero para ello se necesitó un Cray X-MP de 12 millones de dólares conectado a dos ordenadores de un millón de dólares. Este enorme gasto se consideró que valía la pena, ya que permitió crear imágenes que habrían sido difíciles de conseguir en el mundo real.

Toda esta tecnología está muy bien para la gente que tiene acceso al equipo de grabación cinematográfico o de vídeo, a potentes ordenadores y que disponen de tiempo de sobra. Pero la mayor parte de gente que no dispone de ninguna de estas cosas también puede encontrar de gran utilidad los gráficos generados por ordenador. Por ejemplo, las imágenes animadas son una parte importante de toda clase de paquetes de Diseño Asistido por Ordenador (CAD: *Computer-Aided Design*) y todos los buenos juegos de acción dependen en gran medida de representaciones de pantalla atractivas y con una animación bien hecha.

La sofisticación que puede conseguirse en estas imágenes está limitada por la capacidad del ordenador que utilices. El Cray opera a 100 megaflops (100 millones de operaciones de coma flotante por segundo). Pero como las imágenes deben intercambiarse varias veces por segundo para obtener una animación realista, incluso el Cray no puede generar imágenes de calidad lo bastante rápidamente como para conseguir una animación en tiempo real. En vez de ello,



- ANIMACION POR CUADROS
- GRAFICOS POR ORDENADOR
- EXPLICACION DE LOS GRAFICOS PAGINADOS
- UN CUBO MOVIL
- CREACION DE GRAFICOS
- ANIMACION EN TIEMPO REAL
- PERSISTENCIA DE LA VISION
- ANIMACION DE PELICULAS COMERCIALES

las imágenes generadas se filman por separado según un proceso de cuadros convencional.

Los niveles de detalle inferiores de las imágenes permiten que los cuadros se generen más rápidamente. Realmente, existe un simulador de vuelo en el cual las representaciones razonablemente exactas de un avión en vuelo son generadas a razón de 50 veces por segundo, obteniendo efectos animados de gran realismo.

La razón de la dificultad de la animación por ordenador a gran velocidad es la cantidad de información necesaria para cada imagen. Cuanto más detallada sea ésta, más memoria se necesita para almacenarla. Incrementa el número de colores disponibles, y la cantidad de RAM requerida para almacenar la información relativa al color crecerá también.



Cuanta más memoria se utilice para almacenar la imagen, más trabajo tendrá que llevar a cabo la CPU para actualizarla. La razón por la cual la mayor parte de películas comerciales generadas por ordenador emplean la animación por cuadros es simplemente que el procesador no es lo bastante potente como para actualizar grandes áreas de memoria rápidamente. Si la actualización de los gráficos es un proceso lento, incluso para quienes utilizan ordenadores potentes, ¿cómo puede un programador casero obtener animación utilizando un micro? Una solución consiste en utilizar gráficos paginados.

¿QUE SON LOS GRAFICOS PAGINADOS?

Todos los ordenadores domésticos disponen de un área de memoria asociada a la pantalla. Puede tratarse de *memoria mapeada*, lo que significa que a cada posición de pantalla le corresponde una posición de memoria, o puede estar organizada utilizando un fichero de visualización.

Con los gráficos paginados, en vez de construir la siguiente imagen directamente en la memoria de pantalla, se reserva un área en alguna otra parte para dicho propósito. Una vez completa la nueva imagen, se copia a la RAM normalmente asociada a la pantalla. Estas áreas de pantalla adicionales se denominan páginas, por lo cual se conoce a esta técnica como gráficos paginados.

Obviamente, si sólo deseas dibujar una imagen, los gráficos paginados pueden parecer de poca utilidad. Sin embargo, si deseas escribir un programa en el cual una página de texto

vaya seguida de una imagen, piensa cuán conveniente sería que fueras capaz de empezar creando la primera visualización gráfica en algún otro lugar de la memoria mientras el usuario está ocupado leyendo toda la página de instrucciones. Cuando sea necesario el gráfico, se ahorra tiempo, ya que éste se encuentra presente en otra área de la RAM. Si es conveniente, puedes emplear todo el tiempo en hacer cálculos por ti mismo y emplear el ordenador para visualizar las imágenes tan rápidamente como sea necesario.

Pero la ventaja real de los gráficos paginados se consigue cuando deseas visualizar más de una imagen en rápida sucesión. Los comandos gráficos del BASIC habitualmente escriben solamente en la memoria de pantalla. Esto significa que las imágenes se construyen en la pantalla y a continuación se salvan. Mientras que la utilización de gráficos paginados no aporta ninguna rapidez a la construcción física de una imagen en la pantalla, una vez efectuados los cálculos, dichas imágenes pueden transportarse de la memoria a la pantalla muy rápidamente. Así, los gráficos paginados conservan la simplicidad del BASIC, combinada con una velocidad de representación mucho mayor. Y si dispones varias imágenes en distintas áreas de memoria, puedes llamarlas en una secuencia rápida para permitir representaciones relativamente complejas.

ALGORITMO DEL CUBO

Supongamos que deseas construir una secuencia animada simple que muestre la rotación de un cubo. Has decidido que cuatro imágenes serán suficientes para representar una rotación del cubo y deseas que éste gire cinco veces. La estructura de un programa típico podría parecerse a esto:

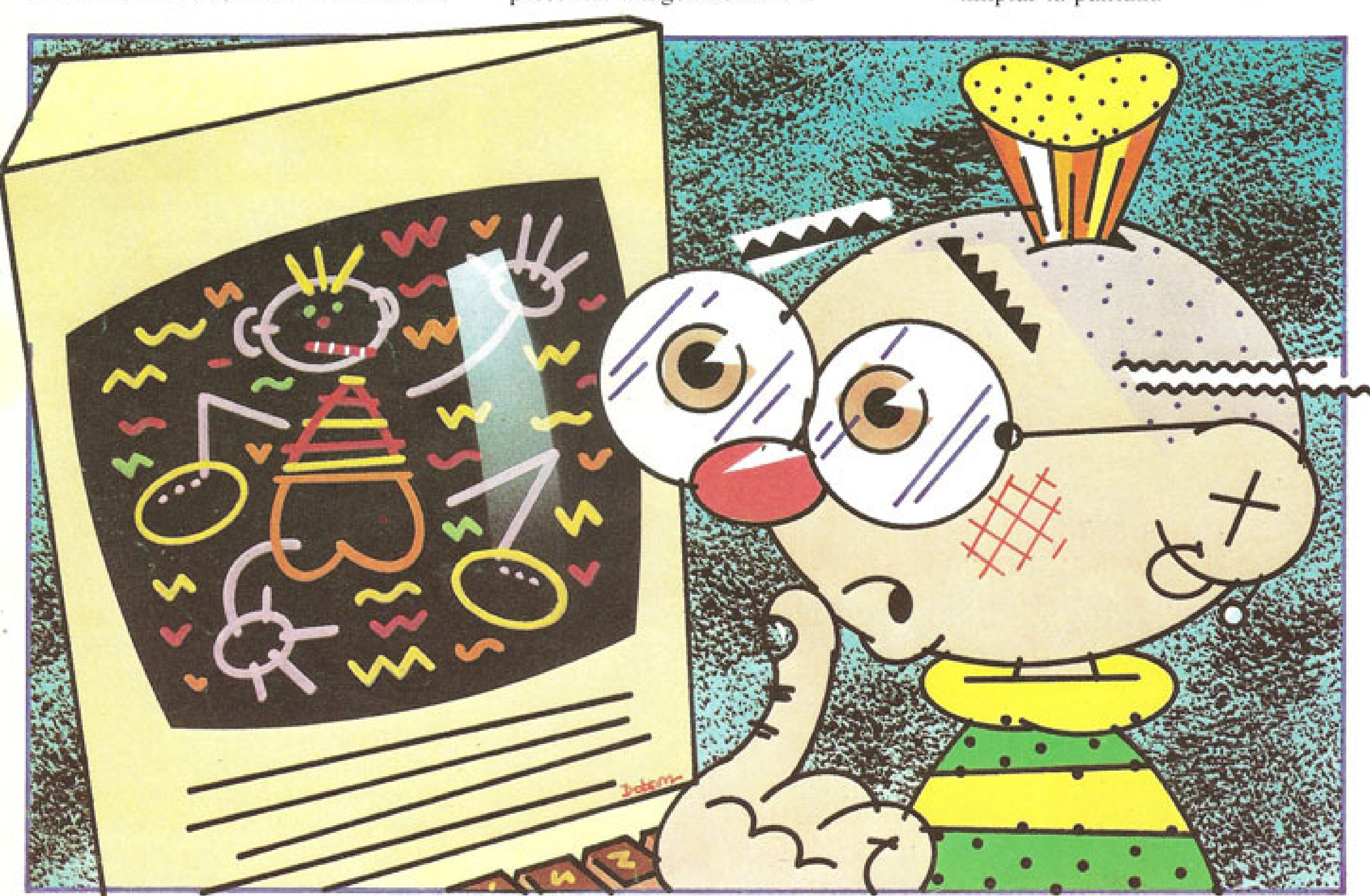
```
for contador = 1 to 5 do
begin
limpiar la pantalla
presentar imagen número 1
limpiar la pantalla
presentar imagen número 2
limpiar la pantalla
presentar imagen número 3
limpiar la pantalla
presentar imagen número 4
end
```

La idea es bastante simple. Se limpia la pantalla, cada una de las cuatro imágenes se dibuja por turno y se repite el proceso hasta que el cubo haya girado cinco veces.

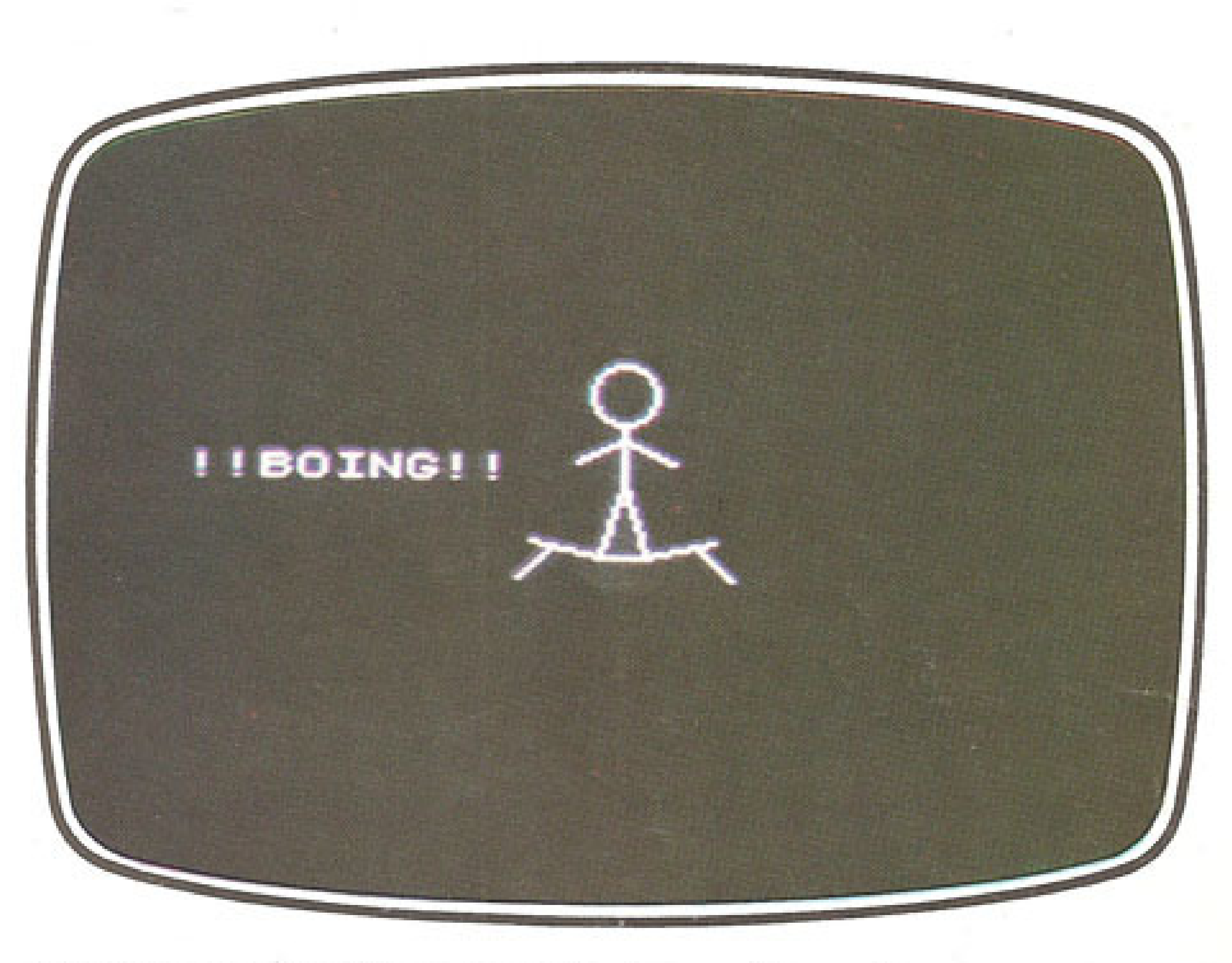
La desventaja de este procedimiento reside en que dibujará cada imagen cinco veces. Los cálculos durarán un cierto tiempo cada vez, por lo que la animación resultante será de baja calidad, con un «salto» apreciable entre cada imagen.

Veamos ahora un algoritmo alternativo, que muestra el procedimiento general de un programa estructurado en torno a las técnicas de gráficos paginados:

```
limpiar la pantalla
```



formar imagen número 1
almacenar datos de pantalla en página de memoria 1
limpiar la pantalla
formar imagen número 2
almacenar datos de pantalla en página de memoria 2
limpiar la pantalla
formar imagen número 3
almacenar datos de pantalla en página de memoria 3
limpiar la pantalla
formar imagen número 4
almacenar datos de pantalla en página de memoria 4
for contador = 1 to 5 do
presentar memoria de página 1 a pantalla
presentar memoria de página 2 a pantalla
presentar memoria de página 3 a pantalla
presentar memoria de página 4 a pantalla
end



La presentación en el Spectrum: un hombre en el trampolín.

El programa es más largo y todavía tienes que pasar por el lento proceso de calcular y dibujar las cuatro imágenes, pero la animación no empieza hasta que se ha realizado dicho proceso.

Una vez almacenadas en memoria, las imágenes pueden volver a llamarse muy rápidamente.

Aunque el dibujo se efectúa todavía en BASIC, el trozo de programa real que mueve una imagen determinada en memoria será una corta rutina en código máquina. Ésta transfiere una pantalla entera de información más de prisa de lo que el ojo puede ver, la esencia de la animación.

DEMOSTRACION DE GRAFICOS

El siguiente programa demuestra una aplicación simple de los gráficos paginados.

Como el programa contiene código máquina para obtener la necesaria velocidad en el intercambio de las imágenes, sálvalo antes de ejecutarlo para evitar percances.

Se te pide que pulses una tecla después de haberse dibujado la imagen y debes pulsar otra tecla más para que comience la alternancia de las imágenes.

Puedes utilizar este programa como base para tus propios experimentos, cambiando las imágenes dibujadas. Un próximo artículo explicará con más detalle las técnicas utilizadas, así como el modo de llevar la sofisticación de la animación hasta los límites de tu ordenador.

Este programa, utilizable en Spectrum de 48K solamente, te dará una animación simple de un hombre saltando en un trampolín. Aunque la mayor parte del programa es en BASIC, hay algo de código máquina que da la velocidad requerida para llamar las páginas desde la memoria.

```
10 BORDER 0: PAPER 0: INK 7: CLS
20 CLEAR 53230
30 GO SUB 220
40 LET srce=64: LET dest=208
50 CLS
60 CIRCLE 128,168,7: PLOT 128,161: DRAW 0,−15: DRAW −10,−10: PLOT 128,146: DRAW 10,−10: PLOT 118,161: DRAW 11,−5: DRAW 10,5
70 PLOT 108,106: DRAW 40,0: PLOT 113, 106: DRAW −8,−8: PLOT 145,106: DRAW 8,−8
80 GO SUB 270: LET dest=dest+16
90 PRINT AT 21,0;"pulsa tecla cuando estes listo": PAUSE 0
100 CLS : CIRCLE 128,141,7: PLOT 128,134: DRAW 0,−15: DRAW −5,−16: PLOT 128,120: DRAW 5,−17: PLOT 118,125: DRAW 10,5: DRAW 11,−5
110 PLOT 108,106: DRAW 15,−4: DRAW 10,0: DRAW 15,4: PLOT 113, 105: DRAW −8,−8: PLOT 144, 105: DRAW 8,−8
120 PRINT AT 6,4;"!!BOING!!"
130 GO SUB 270
140 PRINT AT 21,0;"pulsa tecla cuando estes listo": PAUSE 0
150 LET srce=208: LET
```

```
     dest=64
160 PRINT AT 17,0;"pulsa una
     tecla para RESTORE":
     PAUSE 0
170 FOR n=0 TO 1
180 CLS
190 GO SUB 270: LET
     srce=srce+16
200 NEXT n
210 GO TO 150
220 DATA 1,0,16,17,0,0,33,0,
     0,237,176,201
230 FOR i=53231 TO
     53231+11
240 READ byte: POKE i,byte
250 NEXT i
260 RETURN
270 POKE 53236,dest
280 POKE 53239,srce
290 RANDOMIZE USR 53231
300 RETURN
```

DESCRIPCION DEL PROGRAMA

La línea 10 determina los colores de la pantalla, del borde y de los gráficos; negro, negro y blanco respectivamente. La línea 20 reserva un área de memoria para el código máquina que utilizarás y, a continuación, la línea 30 envía al programa a una subrutina situada en las líneas 220 a 260, que prepara el código máquina. Esta subrutina lee los datos de la línea 220 y los coloca en el área de memoria reservada por la línea 20 antes de volver a la línea 40. Esta última línea define dos variables: ^Gsrce dest. **srce** es el byte alto de la dirección de donde se van a tomar los datos y **dest** es el byte alto de la dirección de almacenamiento temporal. Le indican al ordenador dónde debe leer la imagen de pantalla y en qué parte de la memoria debe colocarla.

Una vez hecho esto, se dibuja la primera de las dos imágenes correspondientes a los dos tiempos, un hombre en el aire impulsado por el trampolín, de lo que se encargan las líneas 60 y 70. La línea 80 envía primero al programa a la línea 270, en donde hay una rutina que pone los números correspondientes a **dest** y **srce** en el programa en código máquina y, a continuación, llama el código máquina para que copie la porción de la pantalla en la que se ve la imagen del hombre con el trampolín.

El paso siguiente consiste en crear la imagen de la segunda página a almacenar. Las dos líneas que lo llevan a cabo son la 100 y la 110, mientras que la línea 120 imprime un efecto de sonido no audible. Esta segunda imagen se almacena en la línea 130, la cual envía al programa a la subrutina situada en la línea 270, mientras que la línea 150 intercambia los valores de **srce** y **dest**, lo cual tiene el efecto de volver a cargar la imagen de la RAM a la pantalla. Las líneas 150 a 210 determinan un bucle que alternará las dos imágenes creadas. Dicho bucle se ejecutará hasta que pulses la tecla BREAK.

Puedes utilizar este programa para preparar tus propias animaciones de dos cuadros, cambiando los comandos gráficos de las líneas 60 y 70 y de las líneas 100 y 110 para obtener nuevas imágenes. Pero en circunstancias adecuadas es posible tener hasta ocho páginas ejecutándose de manera secuencial.

MAPA, CARGADOR Y POKES PARA...

FUTURE KNIGHT (CONT.)

Seguimos con este artículo, la continuación del mapa y comentario del fabuloso juego del FUTURE KNIGHT. El mes pasado vimos cómo pasar fácilmente a través de los innumerables peligros y trampas que nos acechaban, los diez primeros niveles.

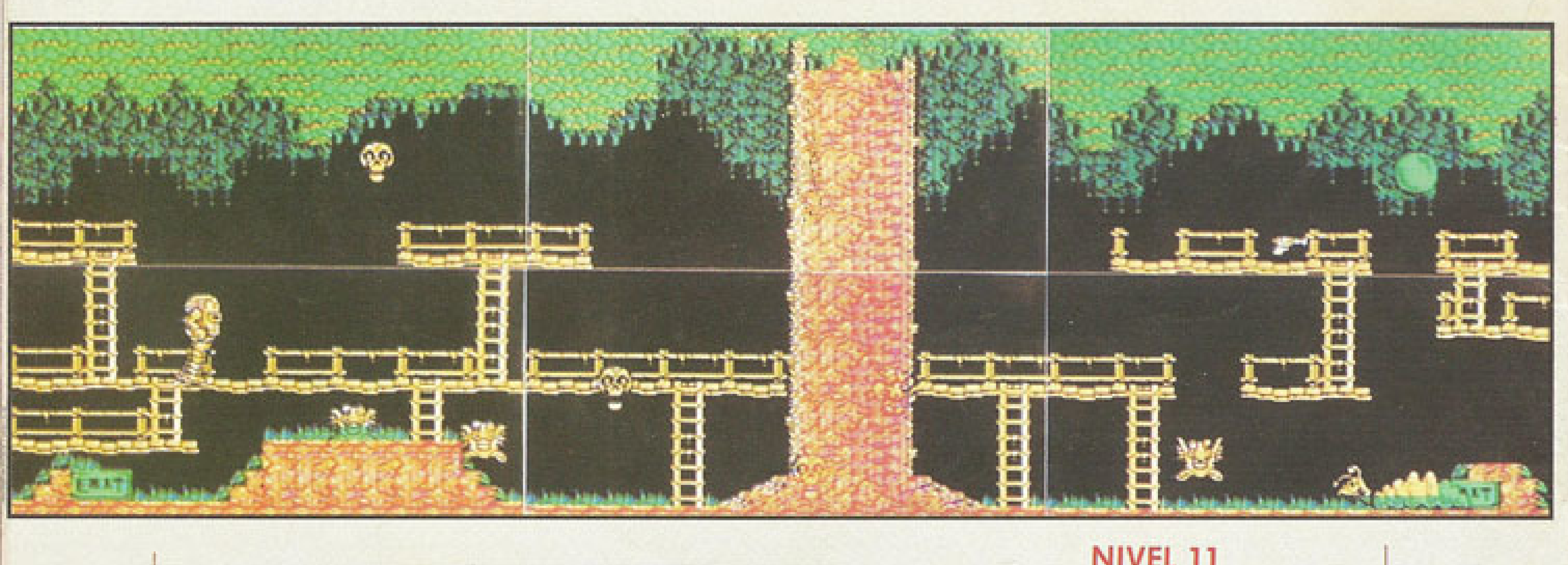
Finalizamos, si recordáis bien o echáis mano a la revista anterior, cogiendo el objeto más importante del juego: el RELEASE SPELL; objeto que no debemos dejarlo o cambiarlo por ningún otro si queremos llevar a buen término la liberación de la Princesa del trágico letargo a la que ha sido sometida por el pérfido y maligno STROC.

... acto seguido pasaremos de nivel (11); en éste tendremos que dirigirnos tres pantallas hacia la derecha, situarnos encima del EXIT y llegaremos al 12. Existen en dicho nivel dos EXIT: el primero de ellos conduce al nivel 15, y el segundo al nivel 13. Para llegar hasta el último, podemos ir por el que queramos, ya que los dos conducen hasta el 14 que es por donde tenemos que ir. Una vez en éste hemos de dirigirnos hacia la derecha y hacia abajo y veremos la salida; nos situaremos encima y pasaremos al 16. En dicho nivel tenemos que dirigirnos 5 pantallas hacia la derecha para encontrar la entrada al nivel 17, tenemos que subir hacia arriba dos pantallas y después seguir hacia la derecha donde encontraremos el pasadizo por el que penetraremos al 18. Para salir de este nivel hemos de dirigirnos dos pantallas hacia la derecha una hacia abajo y encontraremos EXIT, seguimos dos pantallas más hacia la izquierda y veremos la 19.

En este penúltimo nivel sólo tendremos que descender dos pantallas hacia abajo y llegaremos al tan esperado y deseado final. Pero no creáis que todo el monte es orégano. Todavía existen dificultades, terribles dificultades. Para llegar a la princesa avanzamos cinco siniestros pabellones hacia la derecha y vislumbraremos a nuestra muy amada STALINA, pero... ¡Rayos y centellas! nos es imposible estrecharla entre nuestros brazos. ¿Qué hacer? Sencillo. Regresaremos a la pantalla anterior y apretando la tecla correspondiente para activar el RELEASE SPELL podremos rescatar a nuestra amada de las horrendas garras del sueño eterno: nuestra misión habrá acabado con éxito.

Y así es cómo el intrépido «caballero del futuro» cumple su misión: liberar a Stalina.





NIVEL 11

J

PRIMERA PARTE

L

L

K

L

NIVEL 13

K

J

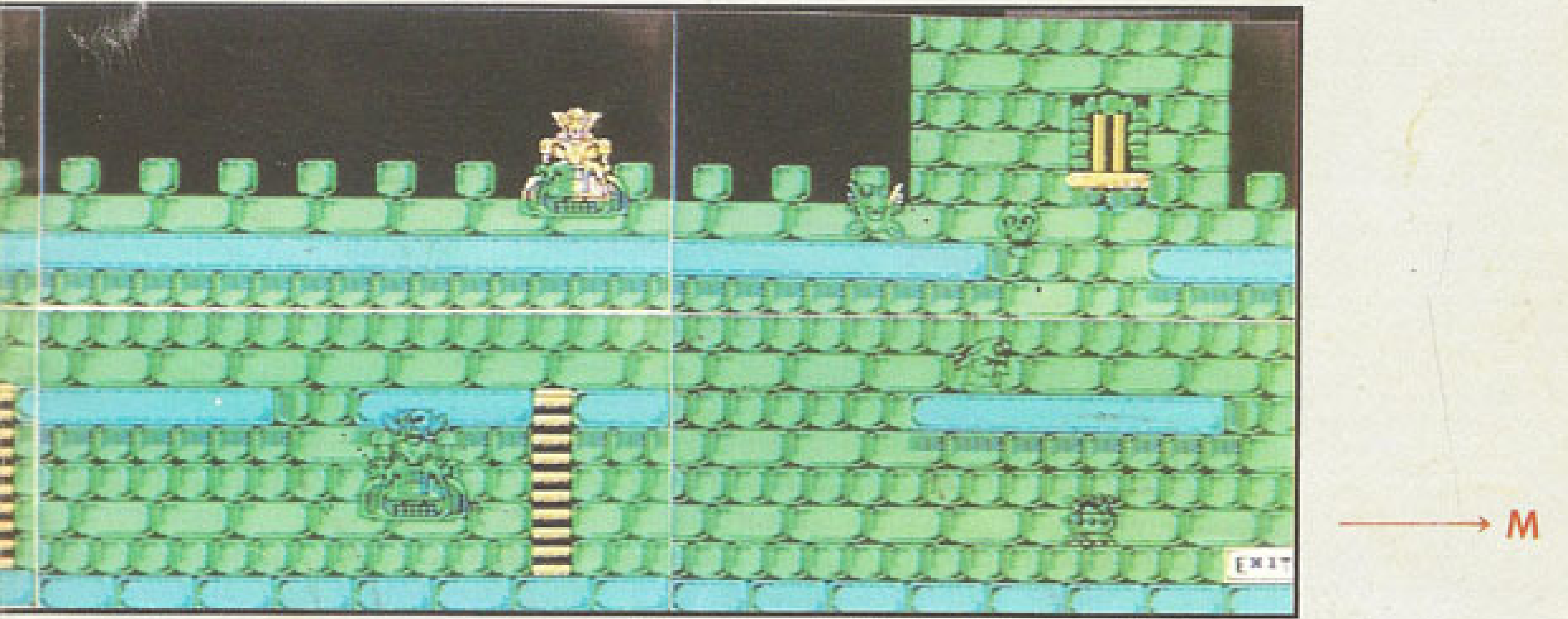
NIVEL 12

M ←

N ←

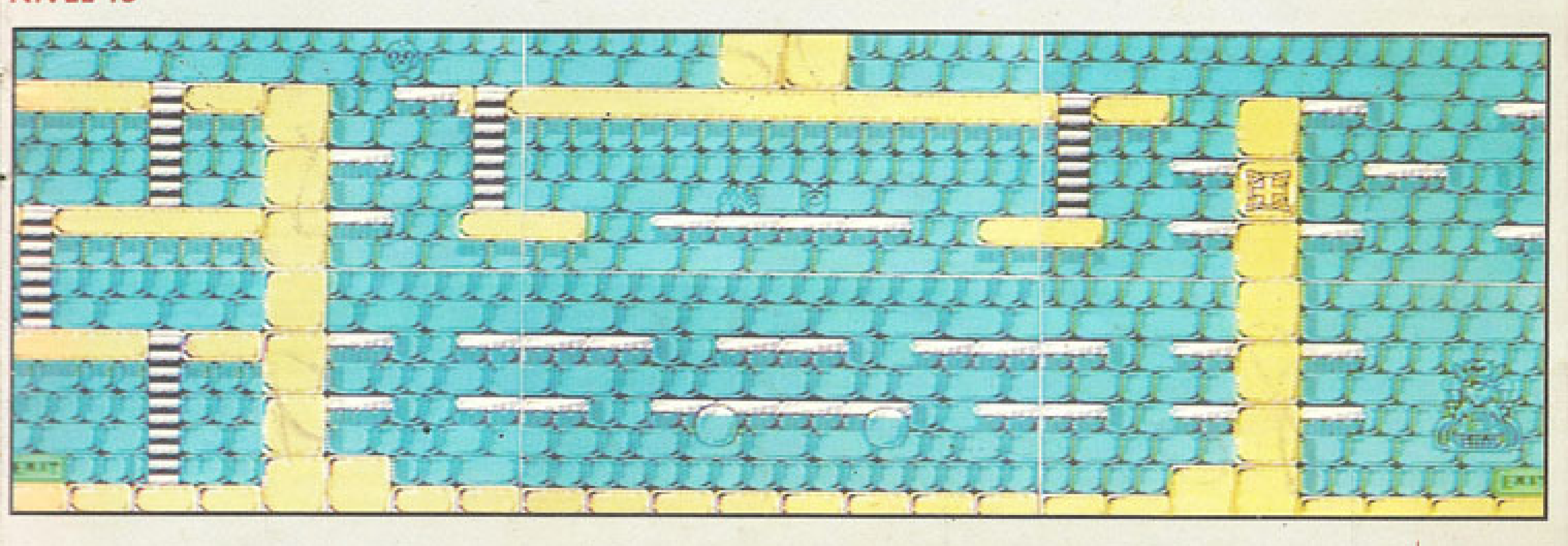
→ O

NIVEL 14

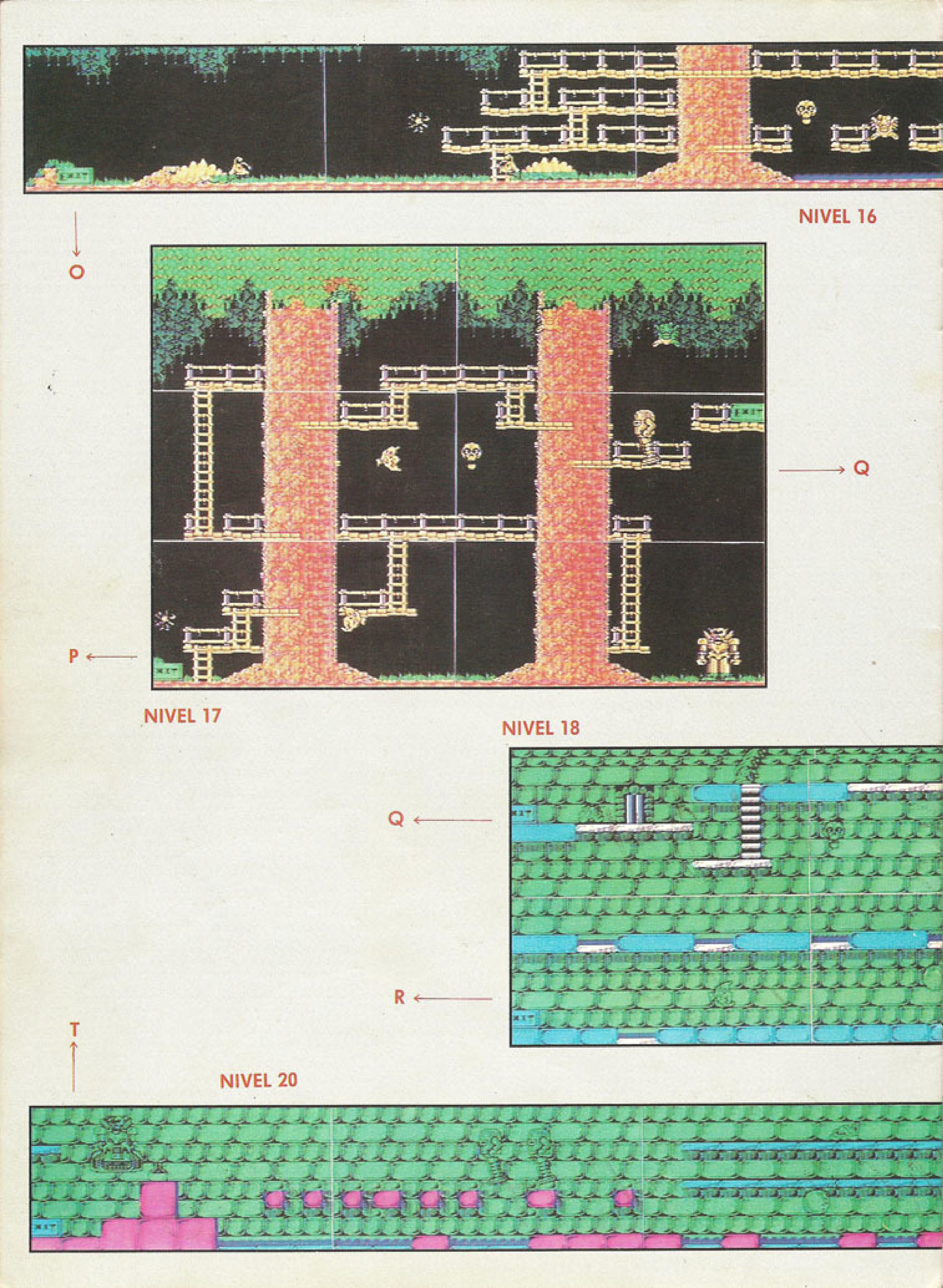


→ M

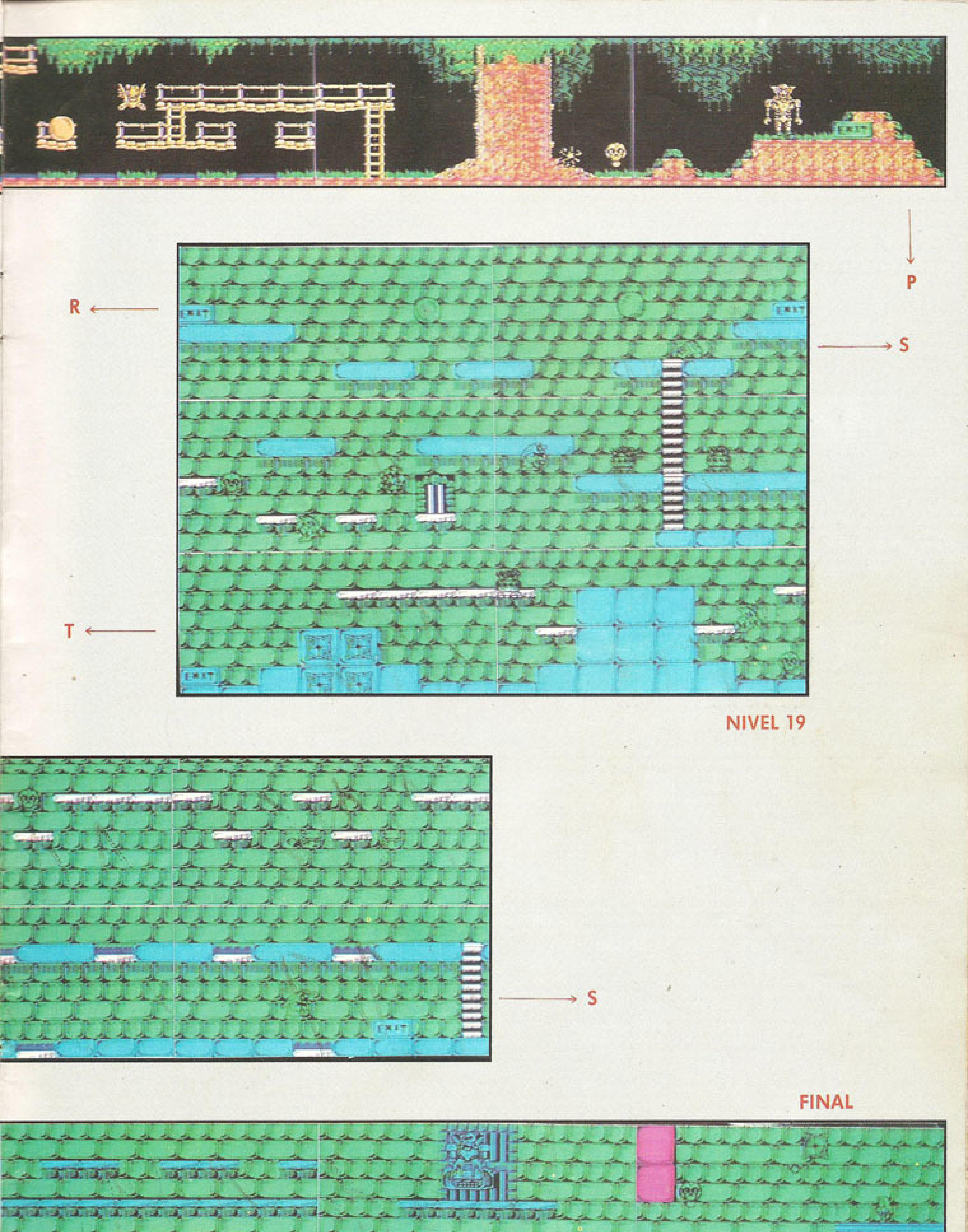
NIVEL 15



N ←



NIVEL 16
O
Q
P
NIVEL 17
NIVEL 18
Q
R
T
NIVEL 20



EXIT
P
R
S
T
NIVEL 19
S
FINAL

TU ERES FANTASTICO

¡Unete a LOS 4 FANTÁSTICOS!

LOS 4 FANTÁSTICOS

AL ESTILO COWBOY

COMICS forum

LOS 4 FANTÁSTICOS

Nº 39

125

COMICS forum

LOS 4 FANTÁSTICOS

¡ATENCION! ¡EL MONSTRUO ATACA DE NUEVO!

¡ANTORCHA, ESPERA! ¡AHORA ME TOCA A MI!

Nº 42

125 PTAS

CADA MES EN TU QUIOSCO

COMICS forum

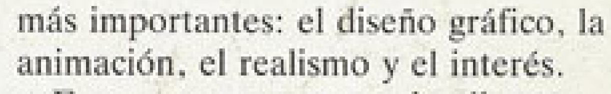
SIMULADORES DEPORTIVOS

Como prometimos en el número anterior, volvemos una vez más con otro monográfico sobre simuladores, dedicado en esta ocasión a los programas de tema deportivo. Debido a su extensión, lo hemos dividido en varias partes, la primera de las cuales os presentamos a continuación:

TODOS SIN EXCEPCION

Cuando comenzamos a elaborar la relación de programas que debían documentar este artículo, nuestro firme propósito era el de reseñarlos todos sin excepción. Sin embargo, después de contabilizar y clasificar hasta cincuenta títulos, no nos quedó más remedio que pasar por alto algunos de los más intrascendentes, de forma que pudiéramos centrarnos con mayor detalle y profundidad en aquellos que verdaderamente pueden interesaros, sin tener que extendernos demasiado.

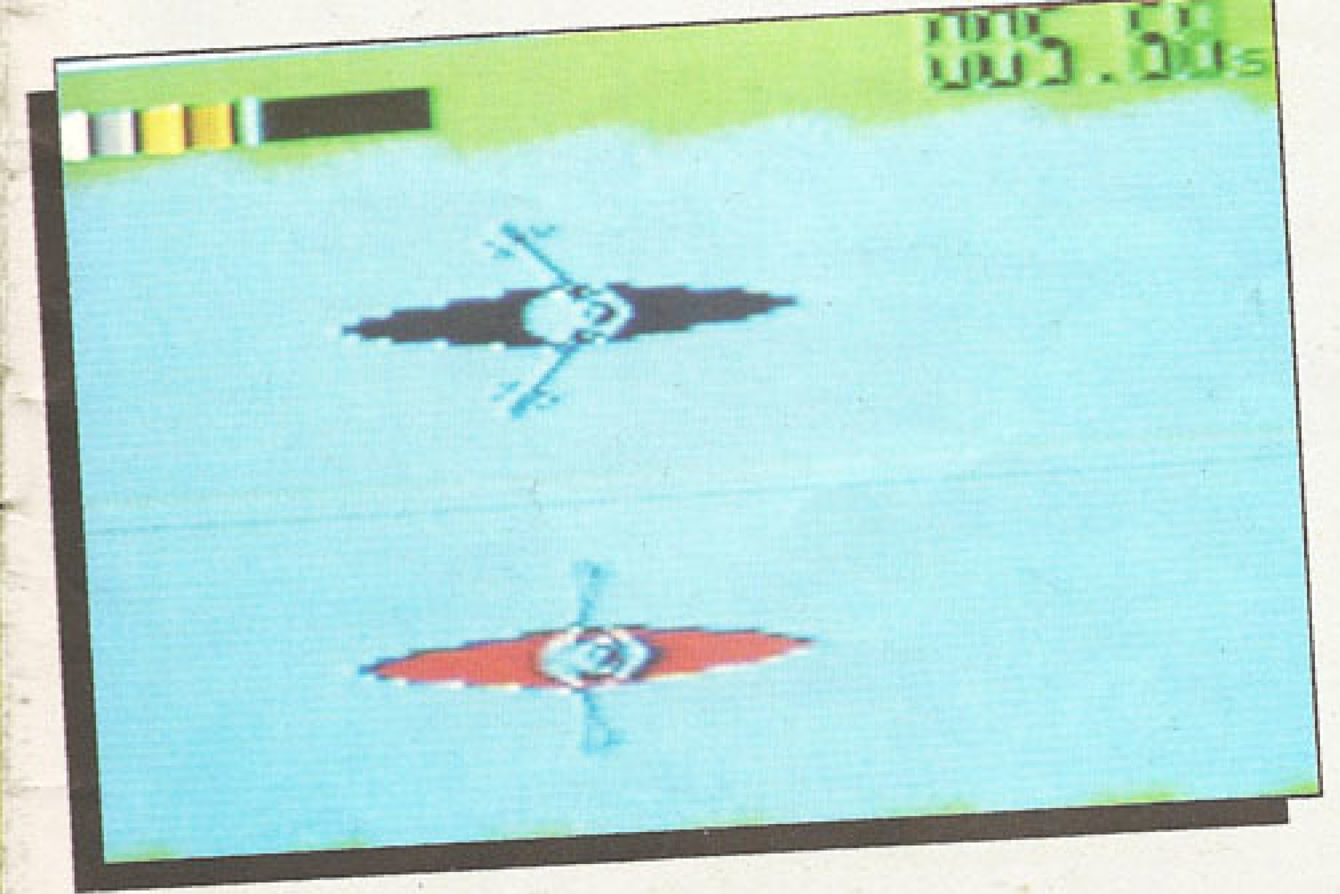
Nuestro objetivo es ofreceros una visión global, en la que tengan cabida todos los simuladores de tema deportivo que por su calidad, su éxito comercial, o por cualquier otra razón, hayan aportado algo a esta breve (pero intensa) historia del software. Para facilitar una panorámica más clara y mejor ilustrada del conjunto, los hemos estructurado en grupos temáticos, calificándolos en tablas comparativas y valorando en ellas los cuatro aspectos que podemos considerar más importantes: el diseño gráfico, la animación, el realismo y el interés.



Esperamos que con todo ello tengáis elementos de juicio suficientes para formaros una opinión sobre la situación actual de los simuladores deportivos, y del camino que hasta ahora han recorrido.

A continuación, comenzaremos nuestro análisis hablando de uno de los grupos más brillantes en la historia de los simuladores: los programas de tema olímpico.

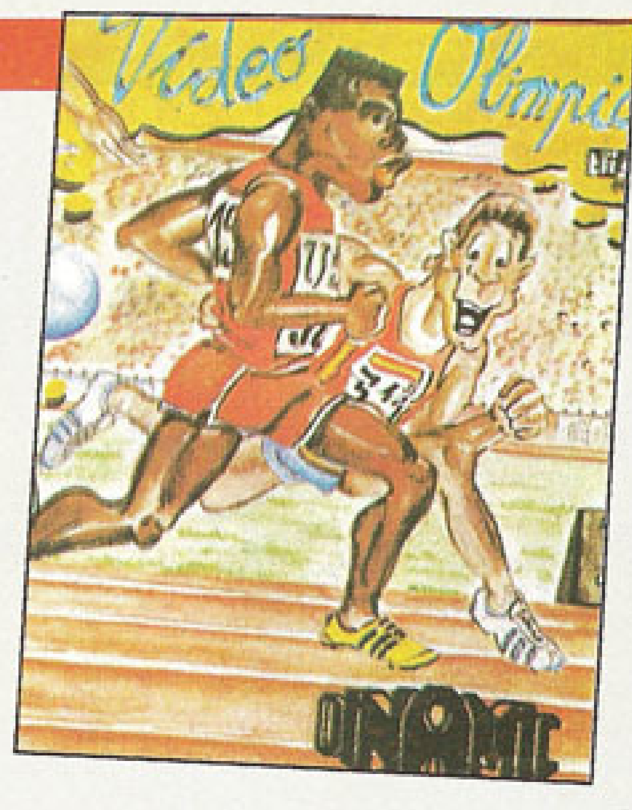
NOTA: No debéis olvidar que algunos de los juegos que vamos a comentar en este monográfico ya no se fabrican, y por tanto es posible que tengáis dificultades para encontrarlos. En todo caso, se suele cumplir la regla de que los programas más interesantes y de mejor calidad son de publicación reciente, y por tanto fáciles de adquirir.



SIMULADORES DE TEMA OLIMPICO

Lo primero que llama la atención en los simuladores de tema olímpico, sobre todo si se trata de los más





recientes, es la espectacularidad de la animación gráfica y el elevado nivel técnico alcanzado en su desarrollo, detalles que quedan claramente reflejados en los éxitos comerciales obtenidos y en la unanimidad de la crítica a la hora de hacer valoraciones. Aunque existen multitud de argumentos que pueden explicar el porqué de este hecho, los más convincentes suelen ser precisamente aquellos que se refieren a la propia naturaleza de los simuladores, como reproducciones o recreaciones de circunstancias de la vida real, en las que se persigue la máxima aproximación posible.

En efecto, la necesidad de ajustarse fielmente a la realidad, ha impuesto a los programadores una continua búsqueda de nuevas y mejores soluciones técnicas, que son las que en mayor medida han impulsado el desarrollo de los programas de simulación.

No obstante, esta ardua tarea de investigación también ha determinado la lentitud con que han ido apareciendo nuevos títulos, con el agravante de la competencia, muchas veces desleal, de otros juegos más fáciles de programar y más comerciales, como son los arcade. Por ello, puede decirse que los simuladores deportivos siempre han ido a la cabeza en cuanto a calidad, pero nunca en cuanto a número.

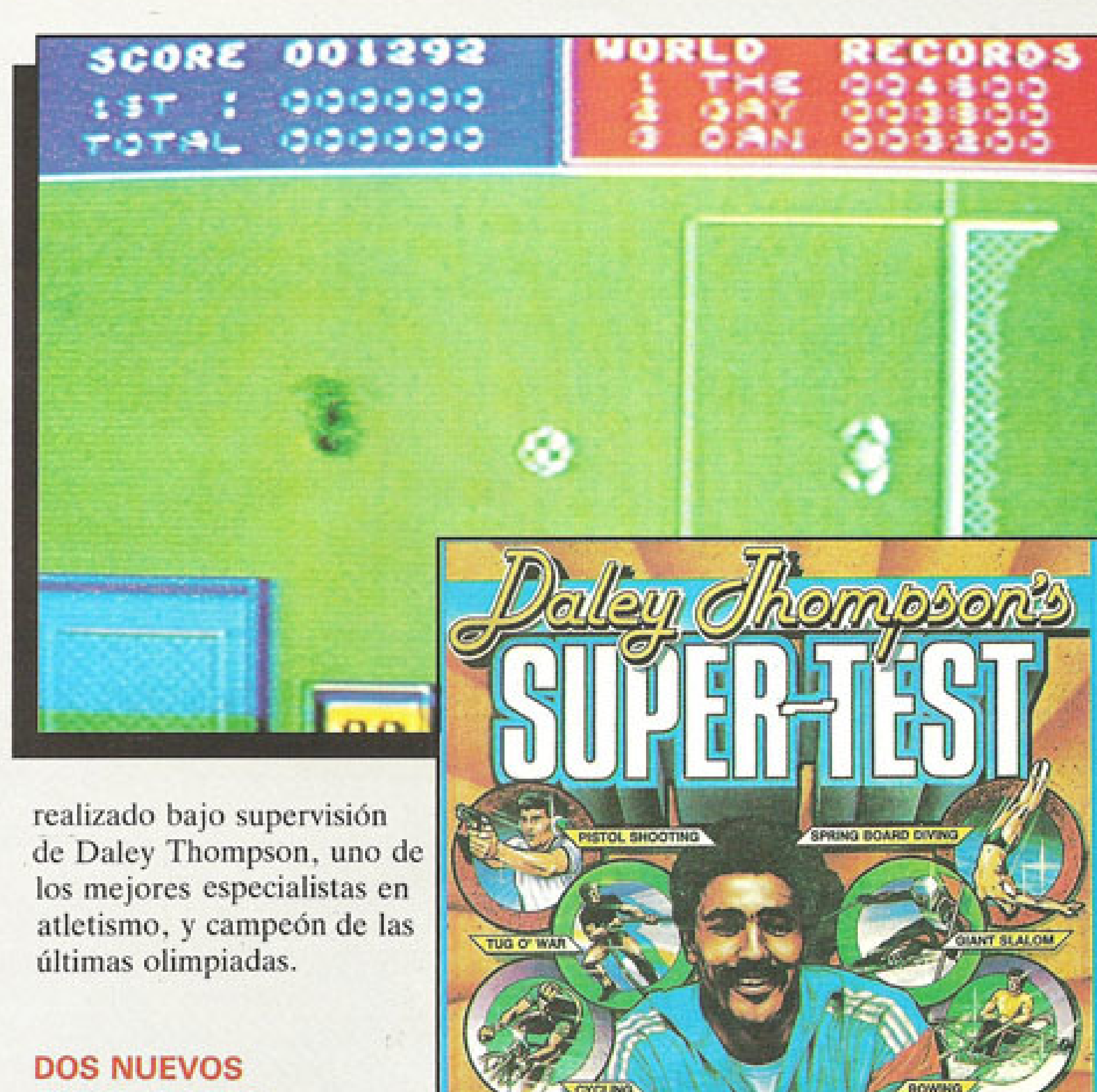
Seguidamente, nos detendremos en seis comentarios pormenorizados de los mejores (y casi únicos) programas publicados hasta la fecha, remontándonos a la primavera de 1984:

OLYMPICON

Este programa fue el primero de tema olímpico que merece la pena mencionar. Con unos gráficos aún algo toscos y una animación deficiente, quedó muy pronto desfasado, aunque logró mantenerse durante un tiempo entre los «número uno» del momento, más por falta de competencia que por sus propios méritos. Fue presentado por la compañía MITEC, ya desaparecida, y se comercializó en España a través de tiendas que importaban directamente desde el Reino Unido.

VIDEO OLIMPIC

Si no nos equivocamos, VIDEO OLIMPIC fue el primer programa de estas características realizado en España, concretamente por la firma DINAMIC. A pesar de su modesta calidad técnica (la animación gráfica se realiza carácter a carácter), todavía hoy se vende en Inglaterra con notable éxito, dentro de una serie de bajo precio muy popular. Las pruebas que reproduce VIDEO OLIMPIC son: 100 metros lisos, jabalina, martillo, 100 metros vallas y natación.

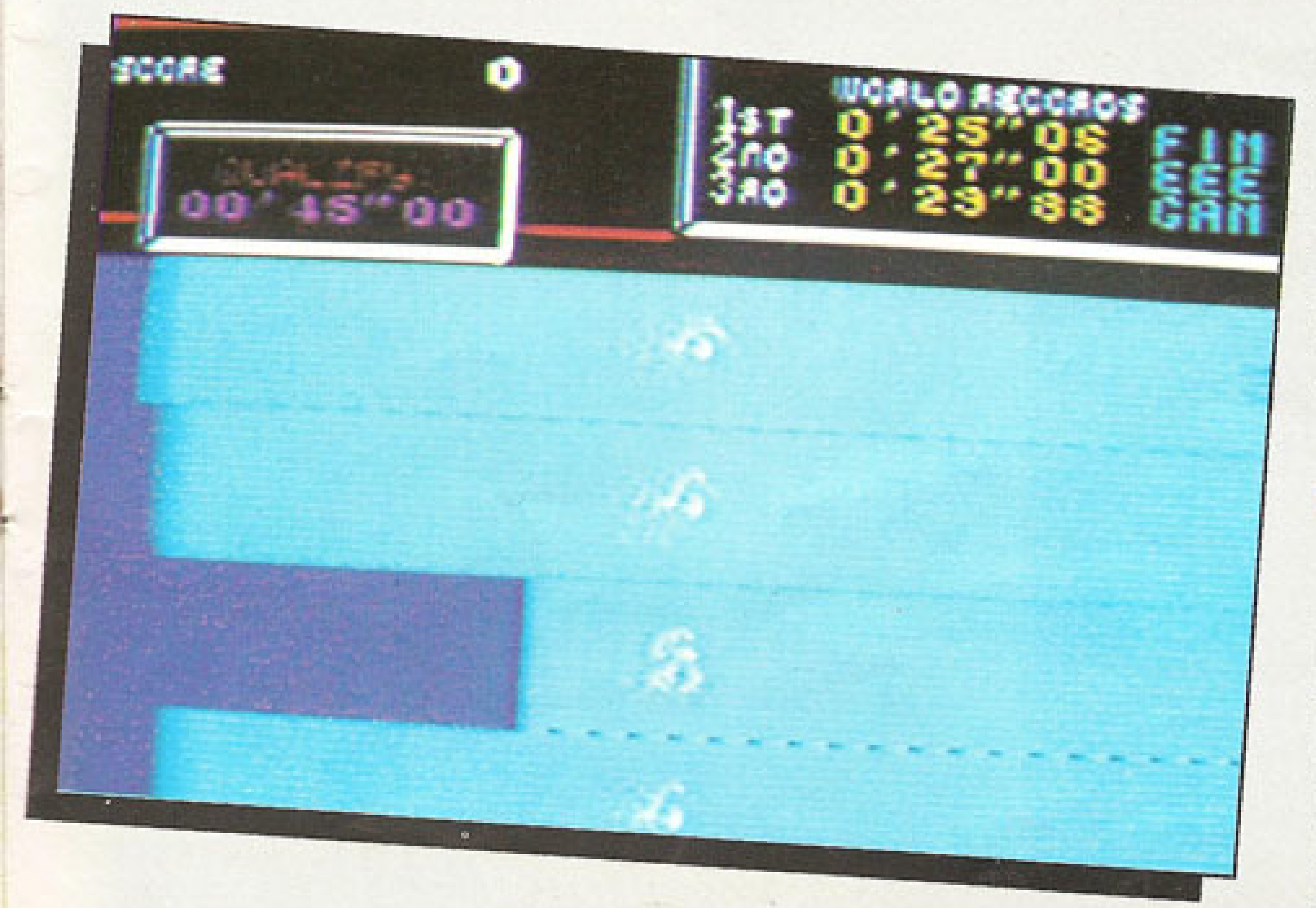


DECATHLON

«Uno de los mitos de la historia del software.» Cayó como un bombazo, avalado por el éxito en las máquinas recreativas, y se vendió como nunca se había vendido hasta entonces ningún otro programa. Fue desarrollado por un equipo de programadores que más tarde alcanzaría una gran fama a través de otros títulos de características similares, y se presentó con el sello de OCEAN. DECATHLON está ambientado en las 10 pruebas tradicionales de la modalidad olímpica del mismo nombre, y fue realizado bajo supervisión de Daley Thompson, uno de los mejores especialistas en atletismo, y campeón de las últimas olimpiadas.

DOS NUEVOS SIMULADORES

Durante 1985, aparecieron dos nuevos simuladores que vinieron a consolidar definitivamente el modelo sentado por el DECATHLON, junto a un gran número de programas deportivos de diversa temática, que más tarde comentaremos. Se trata de los famosos «SUPER TEST» e «HYPER SPORTS», dos sensacionales «números uno» que aún no han sido superados.



SUPER TEST

Este programa se presentó a bombo y platillo como la continuación del DECATHLON, y también llevó la firma de OCEAN. Está considerado como uno de los más completos simuladores deportivos, no sólo por su inigualable calidad gráfica, sino también por el elevado número de

pruebas que incorpora y el nivel de detalle alcanzado en cada una de ellas. Almacenadas en dos bloques de memoria, contiene las siguientes modalidades: tiro con pistola, salto de trampolín, slalom gigante, remo, salto de ski, soga, lanzamiento de penalties y ciclismo.

HYPER SPORTS

Fue realizado por el mismo equipo de programadores que los dos títulos anteriores, aunque esta vez se publicó con producción de IMAGINE. Básicamente, es una continuación del SUPER-TEST, con algunas mejoras en la animación de los atletas y pruebas de mayor interés y originalidad. Las modalidades que incluye son: tiro al plato, natación, tiro con arco, potro, salto de longitud, y levantamiento de pesas.



WINTER GAMES

WINTER GAMES es un extraordinario simulador creado por EPYX, a partir de una versión originalmente desarrollada para COMMODORE. Apareció en 1986, cubriendo el vacío dejado desde la publicación, a finales de 1985, del HYPER SPORTS. Éste es uno de los pocos casos en que la conversión de COMMODORE a SPECTRUM ha sido realizada sin desvirtuar gravemente la presentación gráfica, y solamente por ese detalle ya merece una especial mención. Como su nombre indica, incluye exclusivamente competiciones de invierno: Bobsled, salto de ski, patinaje artístico, patinaje estilo libre, hot dog aéreo, y ski de fondo.

LO ULTIMO

Por el momento, «lo último» en esta clase de simuladores sólo es una realidad para los afortunados usuarios de COMMODORE. Se trata de WORLD GAMES, una pequeña maravilla del software que pronto estará disponible para SPECTRUM. Cabe destacar de este programa, ante todo, la tremenda originalidad de las pruebas que reproduce (pruebas que, por otra parte, ni son olímpicas ni llegarán a serlo nunca): salto de barriles, levantamiento de pesos, rodeo, troncos rodantes, lanzamiento de troncos, salto desde los acantilados de Acapulco...

	DISEÑO GRAFICO	ANIMACION	REALISMO	INTERES
OLYMPICON	7	6	6	5
VIDEO OLIMPIC	8	5	7	7
DECATHLON	9	8	9	9
SUPER TEST	10	9	9	9
HYPER SPORTS	9	10	9	9
WINTER GAMES	10	8	9	8

SPIRITS

● TOPO SOFTO ■ VIDEO-AVENTURA

SPIRITS consigue crear en quien juega con él por primera vez, el efecto de que realmente se está jugando a algo diferente, de que no es una aventura contada como muchas otras, sino algo completamente nuevo:

La pantalla aparece dividida en dos partes. En la zona superior, se desarrolla la acción propiamente dicha, protagonizada por un mago que debe liberar a una princesa y un caballero, para después derrotar al Águila Infernal. En la parte inferior, tiene lugar de forma simultánea el seguimiento de los objetos y los personajes que el mago tiene que encontrar para cumplir la misión con éxito.

La aventura se desarrolla en el interior de dos castillos, unidos entre sí por un pasaje subterráneo. Los pasillos y las mazmorras están ambientados con un buen diseño gráfico, ocupando un total de pantallas muy superior al habitual en otros programas (aunque sin llegar a los 142 habitáculos del SURVIVOR).

En definitiva, una excelente videoaventura a la que podemos augurar un éxito seguro.

EL CARGADOR

```
10 REM CARGADOR VIDAS
   INFINITAS 'SPIRITS'
20 GO SUB 50
30 CLS: PRINT "INTRODUCE LA
   CINTA ORIGINAL"
40 RANDOMIZE USR 23296
50 FOR n=23296 TO 23375:
   READ a: POKE n,a: NEXT n
55 RETURN
60 DATA 205,67,91,221,33,
   203,92,17,223,16
61 DATA 62,255,55, 205, 86,5,
   205,28,93,205
62 DATA 67,91,221,33,203,92,
   17,25,12,62
63 DATA 255,55,205,86,5,62,
   201,50,23,94
64 DATA 33,0,64,17,1,64,54,0,
   1,255
65 DATA 26,237,176, 243, 49,
   95,234,205,196,93
66 DATA 175,50,253,200,195,
   130,187,221,33,0
67 DATA 0,17,17,0,175,55,
   195,86,5,0
```





ANIMACION	9
INTERES	9
GRAFICOS	9
COLOR	8
SONIDO	7
TOTAL	42

SAILING: COMPETICION A VELA

● ACTIVISION ■ SIMULADOR REGATA

Son pocos los deportes que aún no han sido llevados al micro a través de un programa de simulación. En esta ocasión le ha tocado el turno a las regatas, un deporte de élite que



ya puedes «practicar» sin moverte de tu casa.

Como sabes, las regatas suelen consistir en una o varias pruebas de velocidad en un trayecto de forma variable, generalmente triangular, delimitado por balizas, que no pueden ser tocadas por los participantes, computándose los tiempos a efectos de la clasificación final.

Siéntate a la mesa de diseño, dibuja tu propia embarcación a vela de alta competición, y participa con ella en la más emocionante regata internacional, contra las representaciones de 16 países. Ahora tienes oportunidad de demostrar tus aptitudes como capitán de yate, al mando de un rápido velero que tú mismo habrás diseñado previamente.

SAILING ofrece unos buenos gráficos, emoción, y el aliciente de ser el primer simulador de competición náutica disponible para SPECTRUM.

ANIMACION	*9*
INTERES	*7*
GRAFICOS	*8*
COLOR	*6*
SONIDO	*6*
TOTAL	*36*

ENDURO RACER

• ACTIVISION ■ SIMULACION DE RALLYE

La firma ACTIVISION ha presentado recientemente en España la conversión para SPECTRUM de ENDURO RACER, un emocionante simulador de moto-cross que está causando un gran impacto en la crítica especializada británica.

Como es sabido, la característica común de las pruebas motociclísticas de velocidad (entre ellas, el *moto-cross* es comprobar la resistencia de la máquina a su máxima potencia. Son las más espectaculares y, en consecuencia, las que más atraen la atención del público.

El programa reproduce con sorprendente fidelidad las duras condiciones de los rallies todo terreno, a través de cinco circuitos con otros tantos escenarios diferentes. Incluye opción para dos jugadores, teclado redefinible, controles de dirección, aceleración, frenado y salto (más conocido como «caballito»), además de diversos indicadores de velocidad, tiempo, etcétera.

Sobre pistas de arena en el desierto, caminos forestales, barrizales y carreteras asfaltadas, deberás demostrar tu habilidad a los mandos de una sofisticada T.T., en pugna contra un gran número de peligrosos oponentes.



ANIMACION	8
INTERES	8
GRAFICOS	7
COLOR	7
SONIDO	6
TOTAL	36

NEMESIS

● **KONAMI** ■ **ARCADE**

La firma Konami acaba de presentar en el Reino Unido las versiones AMSTRAD, COMMODORE y SPECTRUM.

Este juego, que tanto éxito obtuvo en su versión original para MSX, estará pronto en España, disponible para nuestros ordenadores SPECTRUM.

El juego está planteado según el típico esquema de desarrollo en banda hacia la derecha, con *scroll*, y varias fases sucesivas en dificultad creciente. Sin duda, este modelo no encierra en sí mismo ninguna originalidad, por cuanto se ha repetido en ocasiones innumerables, pero los autores del programa han sabido dotarle de una serie de elementos novedosos, que contribuyen a hacer de NÉMESIS uno de los programas más originales y adictivos de cuantos hemos probado: la imaginación desbordante de los escenarios, la profusión y variedad gráfica, y el emocionante desarrollo de la acción, son detalles que alcanzan cotas de verdadera genialidad.

Allí donde los creadores de NÉMESIS han sabido lucirse con mayor éxito, es en las diversas opciones de la nave protagonista, accesibles desde el teclado según el nivel de bonos de energía obtenidos: dos cápsulas de energía pueden ser cambiadas por un lanzador de misiles; tres permiten doblar el poder de disparo; cuatro, proporcionan un valioso láser, y seis, un campo de fuerza. Coordinar la adecuada selección de las armas con la obtención de cápsulas de energía, esquivar obstáculos, disparos y enemigos, siguiendo siempre hacia adelante, sin poder detenerse, es el difícil objetivo de este frenético arcade. Para facilitarte un poco las cosas, a continuación vamos a describirte brevemente el contenido de sus ocho fases:







El juego comienza con varias oleadas inofensivas de naves en misión de reconocimiento. Destruyendo la última nave de cada oleada, después de haber acabado con todas las demás, aparecerá una cápsula de energía que podrás obtener pasando por encima de ella. Más tarde, llegarás a una serie de grútas defendidas por baterías de cañones y dos peligrosos enemigos a los que habrás de destruir: un monstruo en forma de mariposa, y un curioso personaje que se desplaza sobre un muelle. No dejes que este último se coloque detrás de ti, pues esperará esta ocasión para acabar con una de tus vidas.

Reconocerás el final de la 1.ª fase al llegar a una montaña desde donde salen andanadas de misiles. Después, grandes volcanes que escupen lava sobre su cima tratarán de darte el golpe de gracia, antes de que alcances la fase siguiente. Dos consejos: no intentes destruir los lanzadores de misiles (esquívalos por debajo); acaba con los volcanes subiendo todo lo posible, y disparando varios misiles de baja cota sobre ellos. Por fin, te encontrarás con un obstáculo común al final de todas las fases: la nave nodriza.

Más adelante deberás enfrentarte a grandes imágenes de la Isla de Pascua, alienígenas que se cerrarán sobre ti para devorarte, escenarios invertidos, dragones de roca, y toda una mortífera pléyade de misteriosos enemigos.

Podemos garantizarte que es absolutamente imposible llegar hasta el final sin «ayuda». De hecho, según confiesan en Konami, ni sus propios programadores lo han logrado. Así pues, estamos ante un sensacional programa con un extraordinario reto aguardándote. ¿Te atreves?

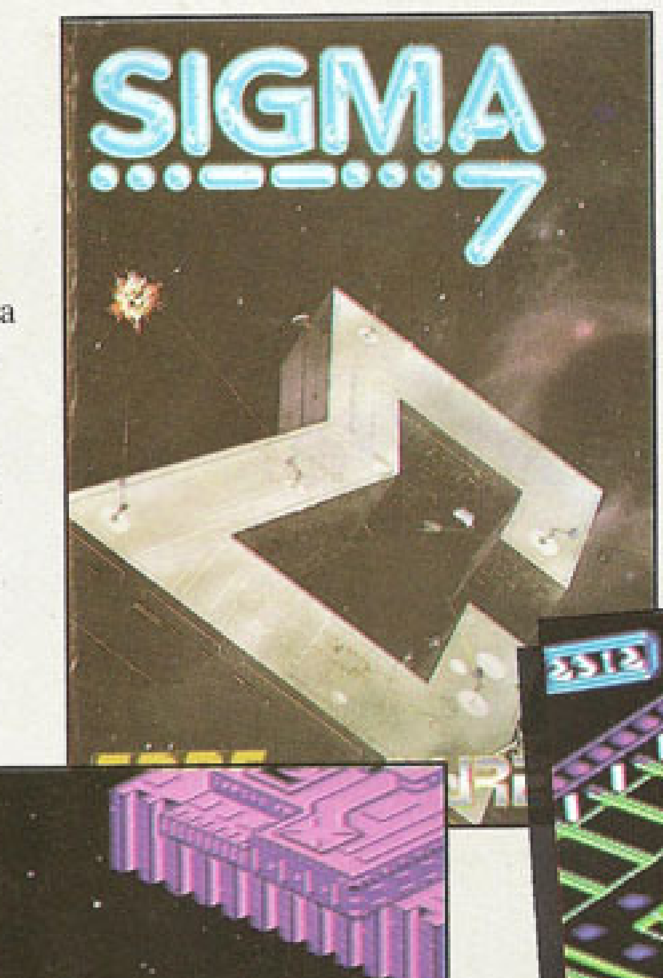
ANIMACION	9
INTERES	9
GRAFICOS	9
COLOR	8
SONIDO	7
TOTAL	42

SIGMA SIETE

● DURELL ■ MULTI-ARCADE

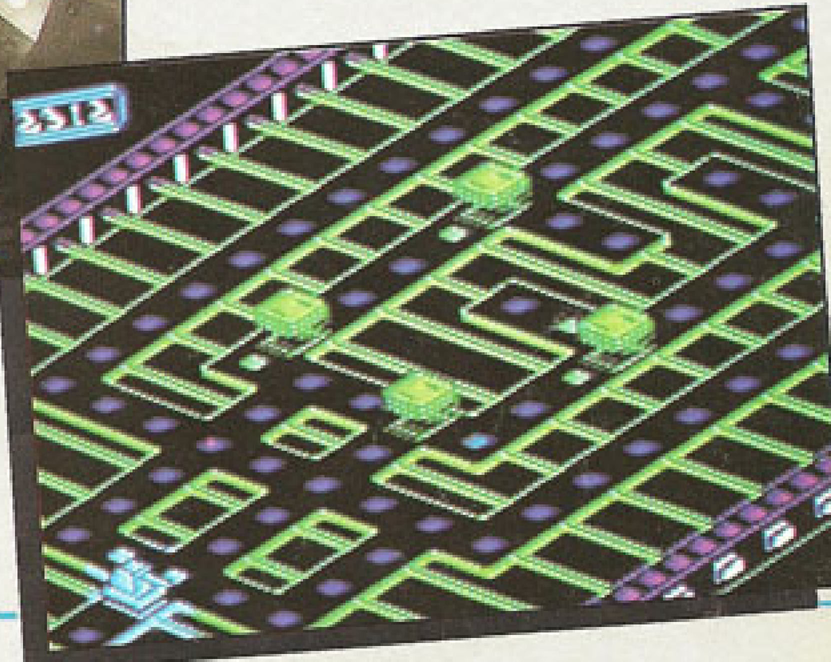
Después de un largo período de silencio, DURELL vuelve al mercado con un extraordinario arcade, ambientado en siete etapas con 3 «subjuegos» diferentes en cada una de ellas: en el primero, deberás volar en una sofisticada nave, luchando contra varias oleadas sucesivas de minas espaciales; en el segundo, tu misión consistirá en recorrer los pasillos de una estación gravitacional, enfrentándote a los robots que la custodian; y en el tercero, habrás de resolver un complejo acertijo futurista.

El juego está planteado con un desarrollo gráfico tridimensional de notable realismo, apoyado en una técnica muy superior a la habitual en la mayoría de los arcades, y una originalidad innegable.



Sin duda, SIGMA 7 es uno de los mejores arcades de la temporada, y creemos que se constituirá en un éxito para Multi-Arcade.

ANIMACION	8
INTERES	8
GRAFICOS	8
COLOR	6
SONIDO	7
TOTAL	37



LOS ARCANOIDES

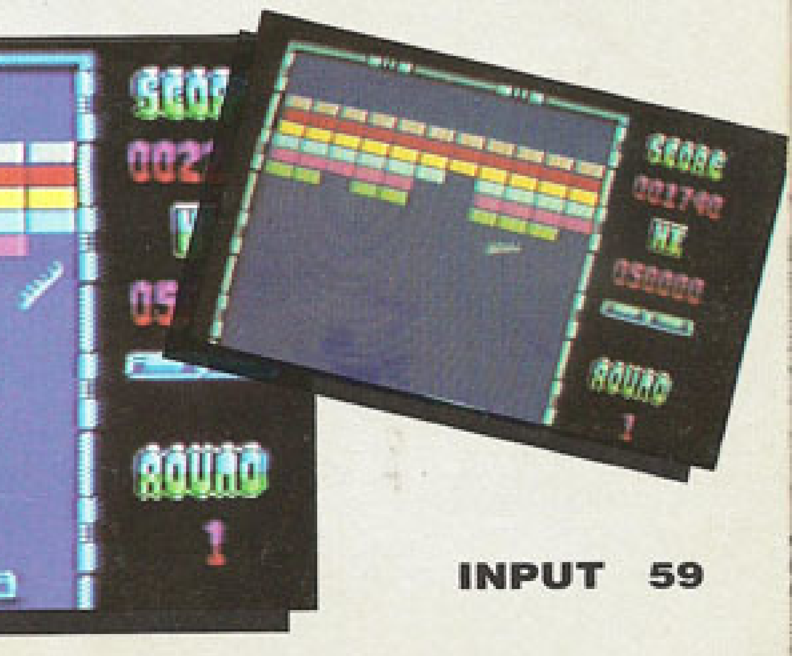
● IMAGINE ■ RAQUETA-ARCADE

ARKANOIDS es una especie de mezcla en la que se combinan dos modelos clásicos para obtener otro extraordinariamente original y novedoso: por un lado, el juego de la muralla de ladrillos, y por otro, el arcade de fases sucesivas, con enemigos inesperados y sorpresas acechando en cada pantalla.

Difícilmente podríamos daros detalles más concretos. ARKANOIDS es un juego que hay que ver para creer, uno de esos arcades en los que ocurren tantas cosas y tan diferentes que apenas se pueden describir.

...Repentinamente, la nave cambia de tamaño, los ladrillos se lanzan contra ella, aparecen tres pelotas en vez de una, cuerpos extraños invaden la pantalla, se invierten las formas, cambian los colores, la nave va ahora más despacio, luego más de prisa... en fin, sólo apto para maestros.

ANIMACION	10
INTERES	9
GRAFICOS	8
COLOR	8
SONIDO	7
TOTAL	42



SHADOW SKIMMER

• SOKTEK-EDGE ■ ARCADE-LABERINTO

Como antes comentábamos, entre los arcades reseñados en este número no hay ningún «patito feo». Al igual que NÉMESIS, SIGMA 7, y ARKANOIDS, SHADOW SKIMMER se caracteriza por una excelente calidad técnica, y unos gráficos de los que antes no se veían más de una vez por temporada.

El objetivo del juego consiste en atravesar los tres sectores del casco de una gigantesca nave nodriza, penetrar en su interior, y allí hacer algo que no sabemos lo que es porque las instrucciones no lo especifican, y porque el nivel de dificultad nos lo ha impedido comprobar por nosotros mismos.

Aunque desde el punto de vista gráfico se trata de un juego realmente bueno, su valoración general puede quedar un poco ensombrecida si tenemos en cuenta otros factores, como el nivel de adicción. No obstante, SHADOW SKIMMER reúne todas las condiciones necesarias para figurar, con la pequeña salvedad mencionada, entre los mejores. Si eres amante de los buenos gráficos, no te lo pierdas.

ANIMACION	9
INTERES	7
GRAFICOS	9
COLOR	7
SONIDO	6
TOTAL	38



LEADER BOARD

• US GOLD ■ SIMULADOR DE GOLF

Por fin tenemos entre nosotros a la esperada versión SPECTRUM de LEADER BOARD, programa firmado por US GOLD, y presentado junto a otro interesante simulador deportivo ya comentado en INPUT, 10 FRAME.

El juego permite competir de uno a cuatro jugadores, elegir palo, calcular distancias, dar «efectos», y dispone de una modalidad de práctica. También es posible seleccionar nivel de dificultad (de amateur a profesional), y elegir número de hoyos.

Antes de realizar un golpe, la visión del campo es analizada minuciosamente por el programa, capaz de calcular la perspectiva correcta desde cualquier lugar donde se halle la bola, dando así una imagen tridimensional de gran realismo. De la misma manera, el efecto del «vuelo» de la bola una vez golpeada, haciéndose más pequeña a medida que se aleja, y proyectando su sombra sobre el césped, está perfectamente logrado.

En suma, un simulador de gran realismo con un nivel gráfico excelente.

ANIMACION	8
INTERES	8
GRAFICOS	8
COLOR	6
SONIDO	5
TOTAL	35

DRAGON'S LAIR (II PARTE)

● SOFTWARE PROJECTS ■ VIDEO-AVENTURA

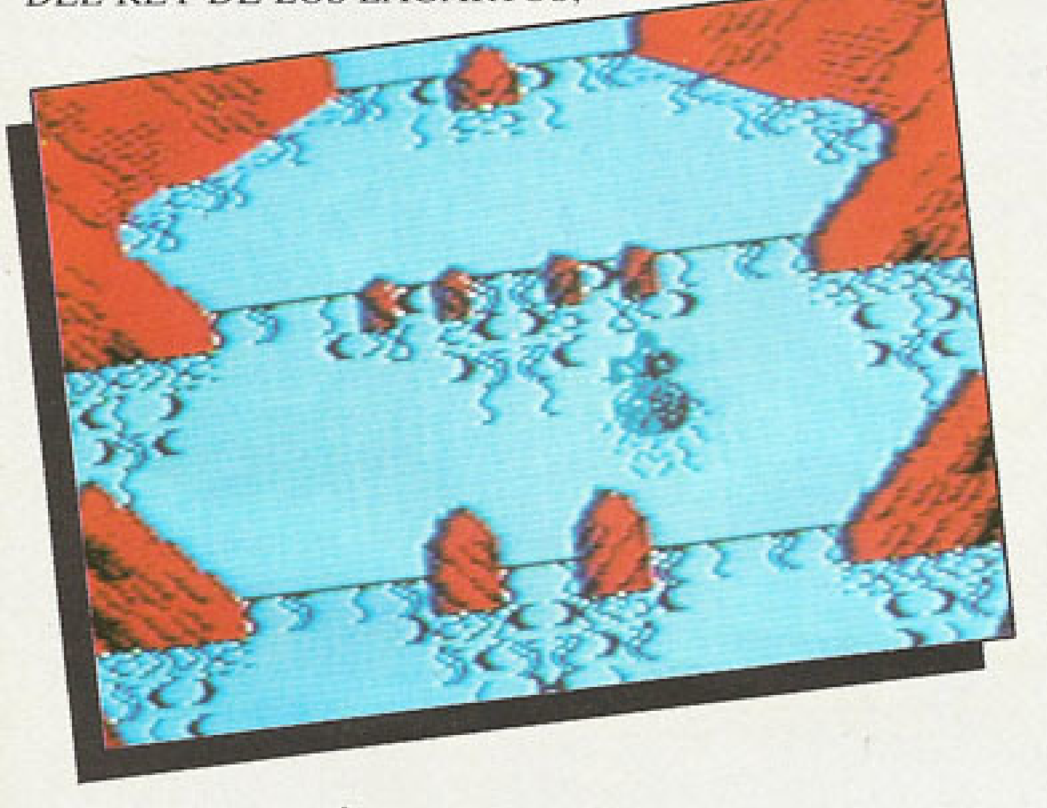
ANIMACION	7
INTERES	7
GRAFICOS	9
COLOR	6
SONIDO	5
TOTAL	34

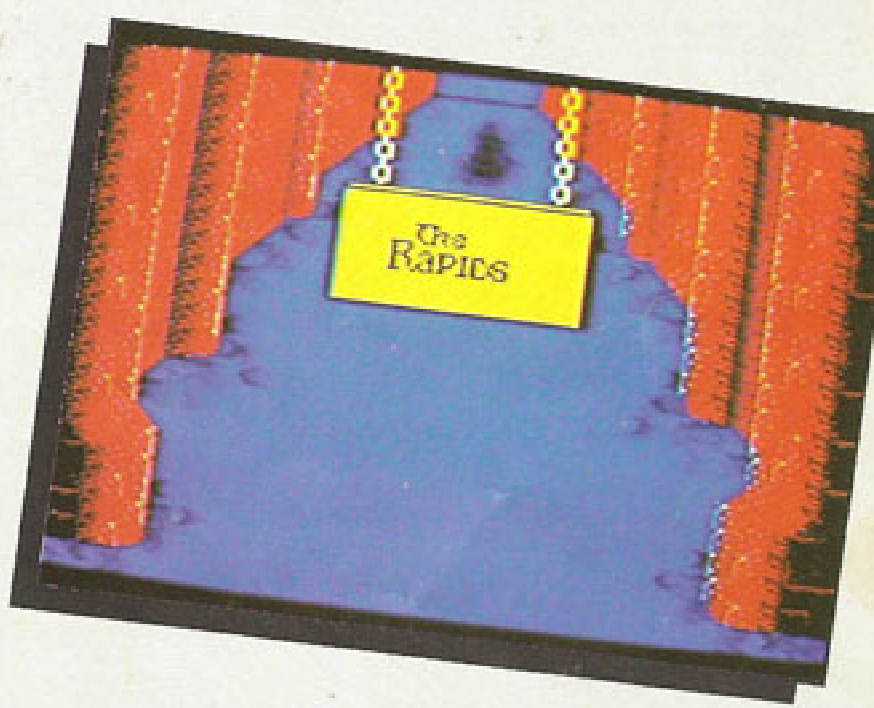
Por mucha habilidad y maestría de que presumas, no podrás evitar que las ocho nuevas pruebas del DRAGON'S LAIR te pongan en ridículo delante de tus amigos.

Seguramente, no tendrás excesivas dificultades para salir con bien de los RÁPIDOS DEL RÍO SUBTERRÁNEO, una prueba facilona para dar cancha a los novatos; pero si además consigues sobrevivir al CAÑÓN DE ROCAS y al SALÓN DEL TRONO, puedes comenzar a creer en los milagros.

De todas formas, poco importa que dures unos segundos más o menos, ya que hagas lo que hagas acabarás igualmente humillado ante la pantalla. No te esfuerces: si no sucumbes en las MAZMORRAS DEL REY DE LOS LAGARTOS,



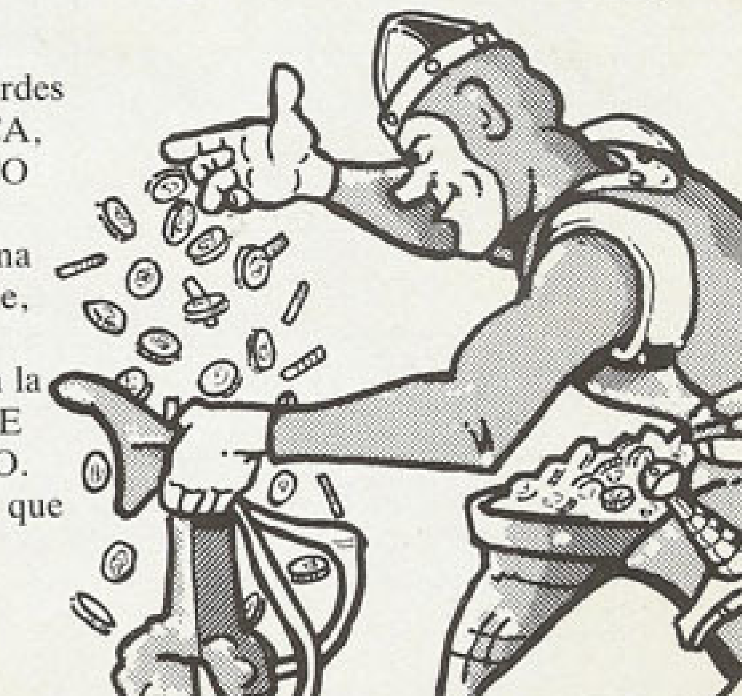




lo harás en el SALÓN DEL CABALLO MÁGICO. Es inútil intentarlo. Las retorcidas mentes de los programadores de SOFTWARE PROJECTS han hecho un buen trabajo, y tú no vas a estropearlo por mucho que te empeñes.

No obstante, si de verdad, de verdad, eres un super maestro especializado en arcades «King-Size», y todavía te queda alguna vida —permítasenos dudarlo—, apostamos la nuestra a que la pierdes en la MAZMORRA ELÉCTRICA, o en el ABISMO DEL MOSAICO MÍSTICO.

Así pues, como llegar a la última prueba es prácticamente imposible, no nos vamos a molestar en mencionar la botella que encierra la clave del juego, ni la GRUTA DE LOS MONSTRUOS DE BARRO. Al fin y al cabo, de poco serviría que lo hiciéramos...



GOLPE EN LA PEQUEÑA CHINA

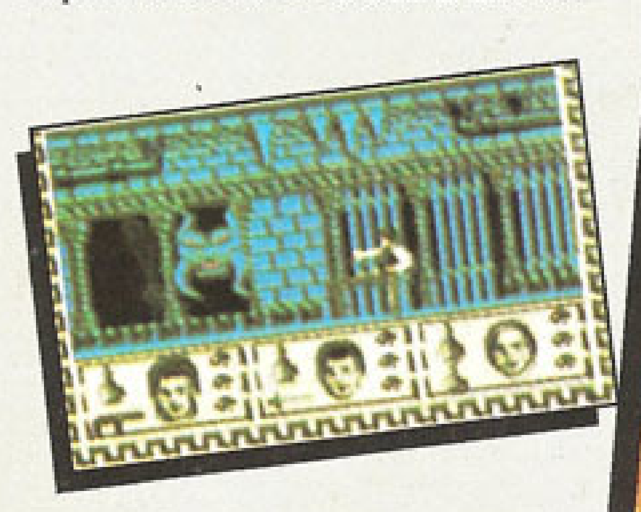
● ELECTRIC DREAMS ■ VIDEO-AVENTURA

BIG TROUBLE IN LITTLE CHINA es el título original de uno de los éxitos cinematográficos de la productora Twentieth Century-Fox, adaptado al micro por ELECTRIC DREAMS. Al igual que la película, el programa lleva en España el título de GOLPE EN LA PEQUEÑA CHINA, que seguramente os sonará mucho.

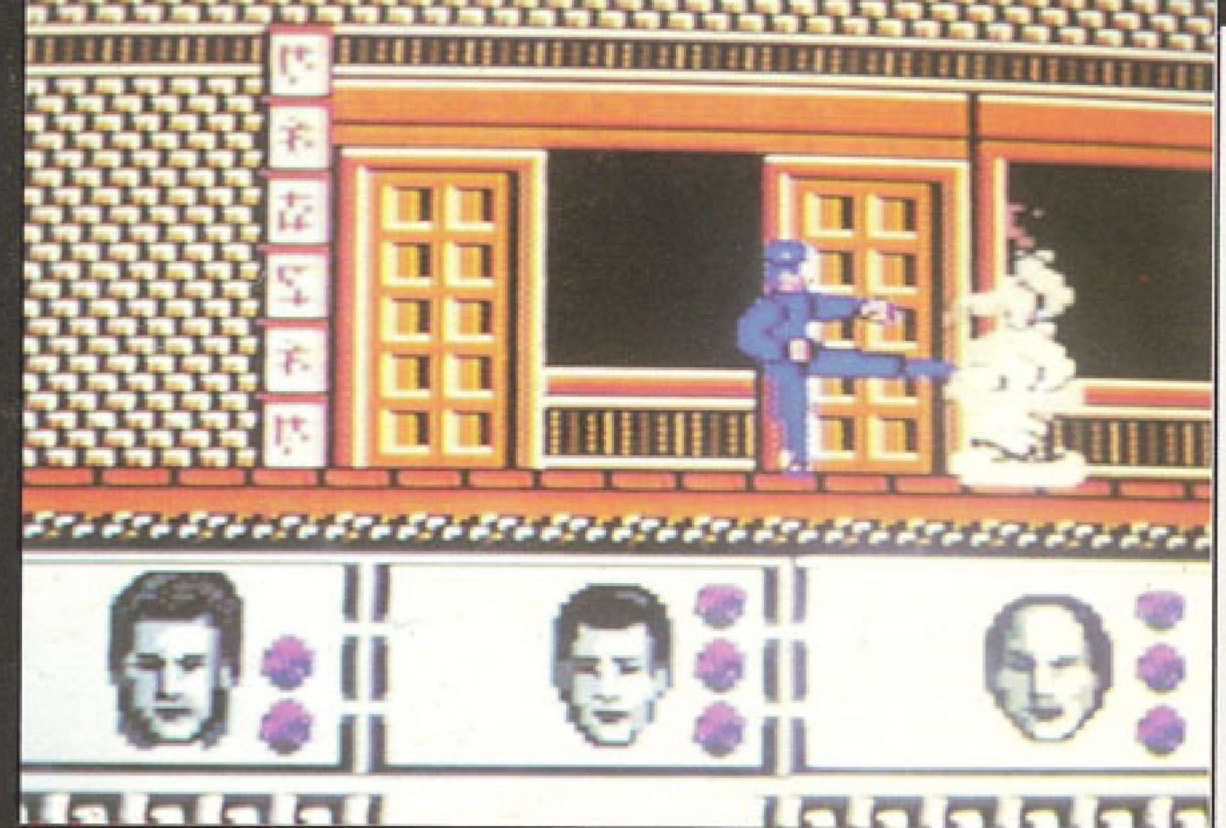
Como ya sabréis los que hayáis visto la película, los protagonistas de la aventura son Jack Burton y Wang Chi, dos jóvenes que, ayudados por el mago Egg Shen, tienen que infiltrarse en los dominios subterráneos del Barrio Chino, y liberar a sus respectivas novias, que han sido capturadas para hacer con ellas sacrificios humanos.

Los tres personajes se mueven juntos por la pantalla, pero tú sólo manejas a uno de ellos (a tu elección), mientras los otros dos le siguen a todas partes (este sistema ya lo habíamos visto en sendos programas de ASTERIX y EL MISTERIO DEL NILO). Cada uno es especialista en un tipo de combate distinto: Jack Burton maneja los puños y, si la encuentra, una mortífera pistola Bushmaster; Wang Chi, además de ser experto en Artes Marciales, también usa la espada. Por último, Egg Shen combate con poderosos rayos mágicos.

Desde el punto de vista técnico, no cabe duda de que GOLPE EN LA PEQUEÑA CHINA supera a la mayoría de las apresuradas adaptaciones de películas que se han hecho hasta ahora. Aunque los gráficos son modestos, el desarrollo original de la película y su emocionante argumento han sido trasladados al micro con gran acierto, haciendo de este programa una interesante y adictiva videoaventura.









ANIMACION	*7*
INTERES	*8*
GRAFICOS	*7*
COLOR	*7*
SONIDO	*7*
TOTAL	*36*

SURVIVOR

• TOPO SOFT ■ AVENTURA ECOLOGISTA-FUTURISTA

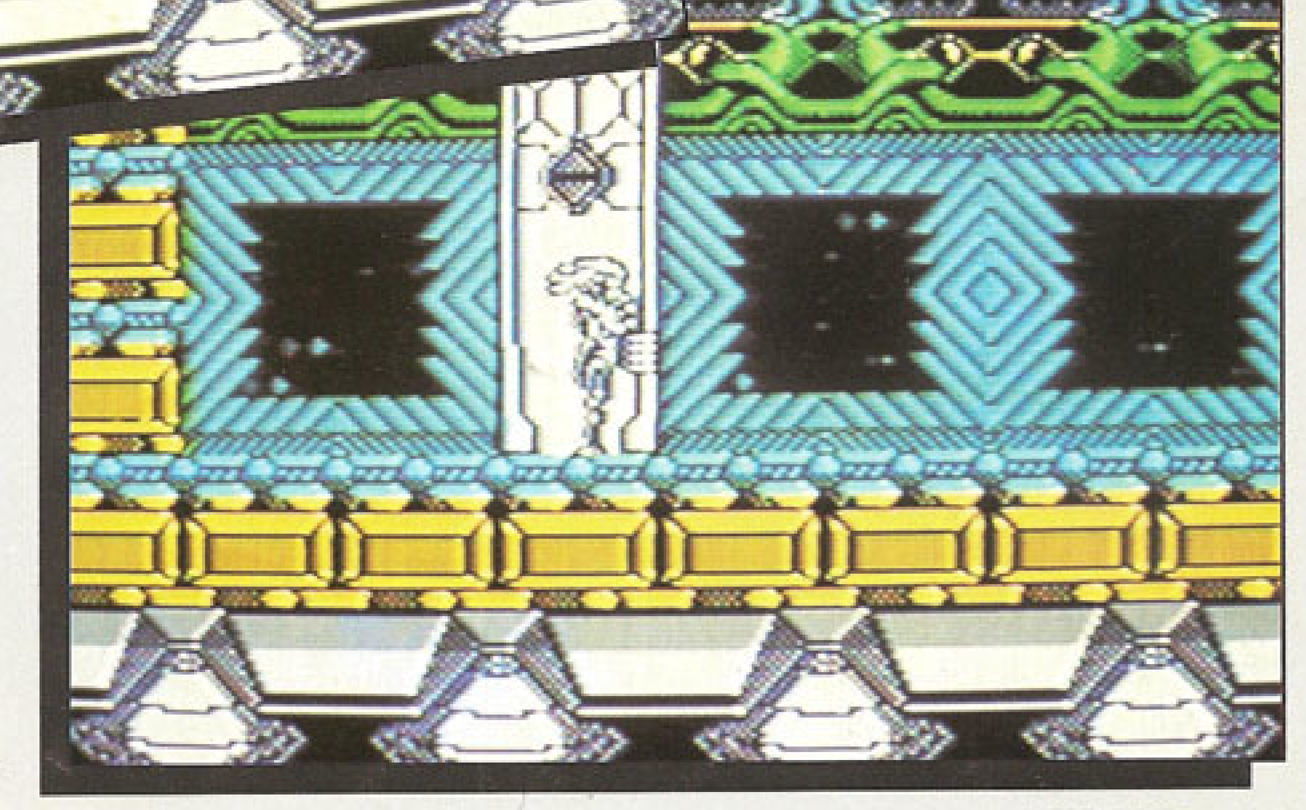


El argumento de este programa (sin duda el mejor de cuantos ha presentado TOPO), es una especie de alegato ecologista en versión macabra. El objeto del juego consiste en ayudar a un espantoso bicharraco encinta (es decir, embarazado) a sobrevivir y perpetuar su especie en el interior de una gigantesca nave-reserva, tripulada por unos pequeños seres extraterrestres. Para lograrlo, deberá llevar una dieta equilibrada a base de tripulantes humanoides en crudo, y poner en lugar seguro los diez huevos de donde saldrán las futuras crías, aparte naturalmente, de superar multitud de peligros.

Como puedes apreciar, la finalidad «ecológica» de este programa es opuesta a la del famoso ALIENS, en que el jugador hacía todo lo posible por exterminar a la odiosa madre de los monstruos y su progenie.

Estamos completamente seguros de que un argumento tan original y adictivo como éste, con unas pantallas como las que podéis ver en las fotografías, tiene garantizado el éxito.

ANIMACION	9
INTERES	9
GRAFICOS	9
COLOR	8
SONIDO	8
TOTAL	43



CORTOCIRCUITO

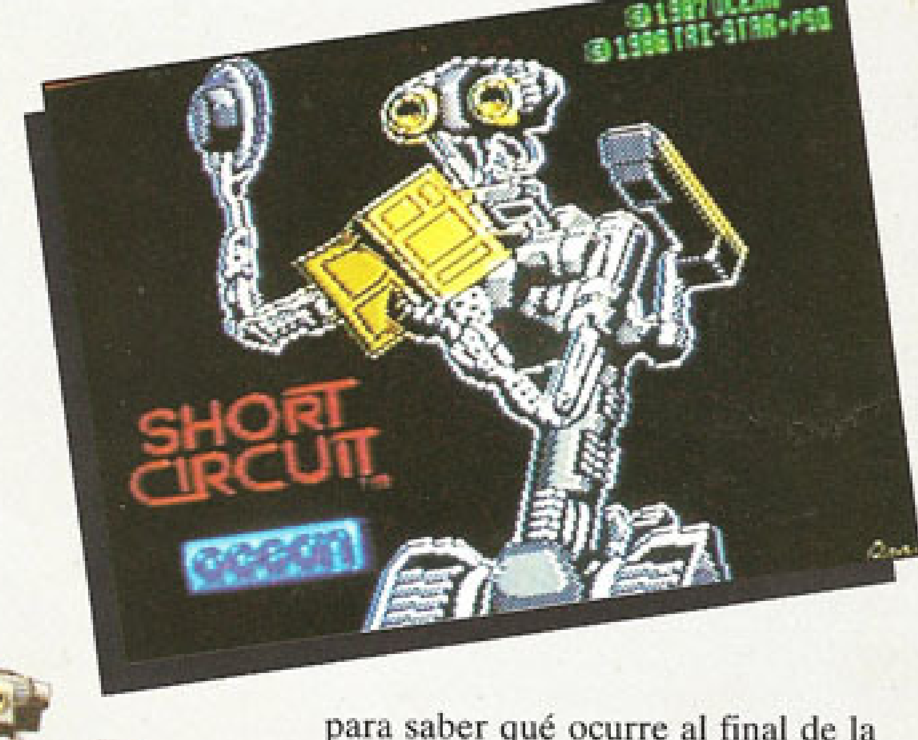
● OCEAN ■ VIDEO-AVENTURA

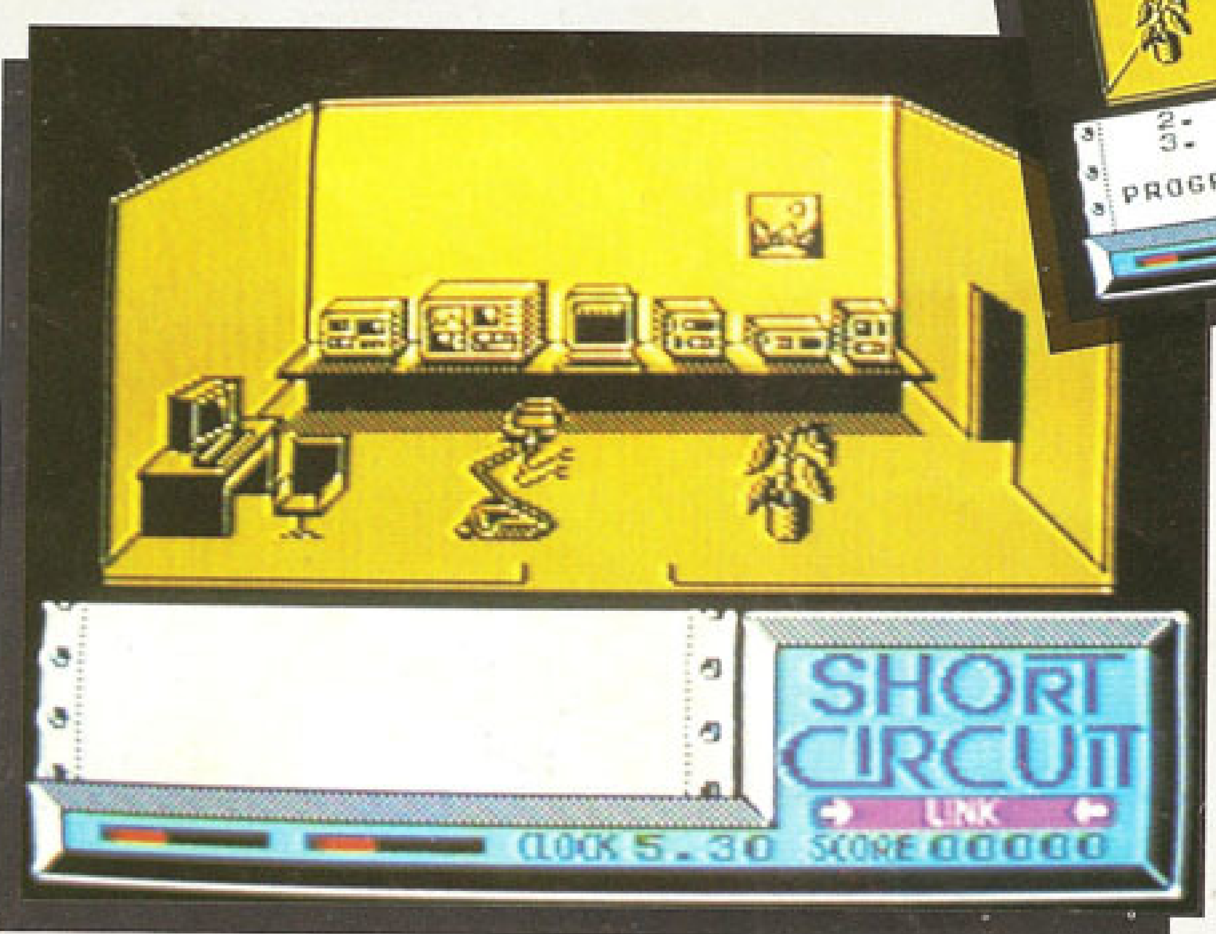
«¡Hay que capturarlo antes de que sus armas maten a millones de inocentes!», bramó el presidente. «¡No! Hagámoslo explotar cuanto antes», contestó el jefe del departamento de seguridad, pensando que sólo así podría irse a cenar tranquilamente a su casa...

Había ocurrido lo impredecible: un rayo activó los circuitos del robot experimental número cinco de la serie SAINT, haciéndole cobrar vida, y dándole la capacidad de raciocinio suficiente como para saber que iban a por él, y que tenía que hacer todo lo posible por impedir la captura. Dentro de la fábrica donde lo habían almacenado, buscó los componentes necesarios para activar su láser y su mecanismo de super salto, con ayuda de las terminales del ordenador central. Después, tendió varias trampas a los guardias de seguridad, y cargó en su memoria los seis programas necesarios para ampliar sus funciones, antes de escapar precipitadamente de allí, iniciando la segunda parte de su singular aventura.

Una vez en el exterior, se vio envuelto en una desenfrenada carrera, perseguido por guardias y robots de seguridad, y hostigado por animales salvajes, en un paisaje lleno de obstáculos. Más tarde...

Mucho nos tememos que para saber qué ocurre al final de la segunda fase de SHORT CIRCUIT, vas a tener que esforzarte por descubrirlo tú mismo. Teniendo en cuenta que el nivel de dificultad no es muy alto, y que el juego resulta extraordinariamente adictivo, no tendrás muchos problemas para lograrlo, aunque de todas formas tampoco va a ser demasiado fácil. ¡Ánimo! A ver cuánto tardas en capturar el robot número cinco.

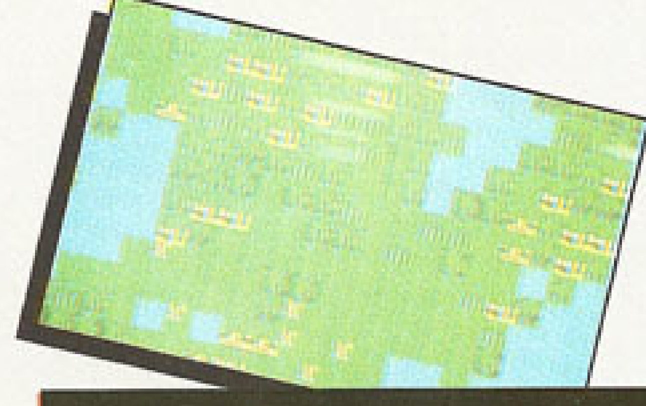




ANIMACION	5
INTERES	8
GRAFICOS	8
COLOR	8
SONIDO	7
TOTAL	36

EXPLORER

● ELECTRIC DREAMS ■ VIDEO-AVENTURA ESPACIAL



ANIMACION	9
INTERES	7
GRAFICOS	8
COLOR	8
SONIDO	7
TOTAL	39

Te encuentras a 30 billones de años luz del taller espacial más próximo, y todavía más lejos de la tienda de aeronaves usadas donde te dieron el timo. ¿Que cuál es el problema? Tu nave no ha podido resistir la entrada en atmósfera de uno de los mundos del «área exterior», y varios trozos de sus partes más vitales se han desprendido del casco, perdiéndose en algún lugar de las vastas extensiones de jungla del planeta. Maldecir al vendedor que te estafó un millón de créditos, un tal Billy «el honesto», no servirá de nada, como tampoco te será de mucha ayuda lamentarte de tu desgracia. Lo mejor que puedes hacer es coger tu modesto equipo de búsqueda, compuesto por un jet portátil, nueve aerofaros de radio, diez zumbadores



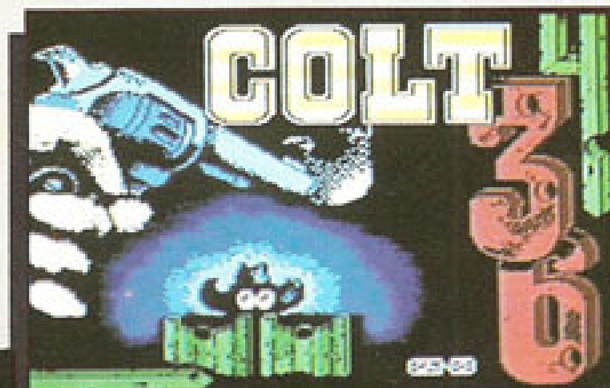
antigravedad, un sonar de objetos, un radiovector, una brújula, una pistola láser, y unas botas pesadas (¡modesto!), y encontrar los componentes que te permitirán reparar la nave y regresar a la

Fácil.

COLT 36

● TOPO SOFT ■ ARCADE WESTERN

Armado con un revólver del calibre 36 largo, y ayudado por un individuo que te indica por dónde van a salir tus enemigos (un tipo sordomudo con una intuición extraordinaria, que te hace señas con la mirada), deberás limpiar un poblado del viejo Oeste de bandidos, pistoleros, indios, ventanas, botellas, velas, pájaros, caballos, y todo lo que se te ponga por delante, sea inocente o culpable (ya sabes, disparar primero y preguntar después), como si tú fueras el mismísimo Billy el Niño.



El programa ofrece un completo panorama del poblado, visto desde el lugar donde te encuentras, con un punto de mira en el centro que indica el lugar hacia el que apuntas. Las municiones están limitadas a cuarenta disparos en cada fase, pero cada vez que superas un nivel del juego, acabando con todas sus bandas, obtienes un nuevo cargador de otros 40 disparos.



ANIMACION	8
INTERES	8
GRAFICOS	8
COLOR	7
SONIDO	7
TOTAL	38

EL ZOCO



Intercambio/compro/vendo programas para el Spectrum Plus, además compro MODEM para dicho ordenador. Mandar ofertas a: Antonio Toribio Carreras. Polígono Puerta Madrid, sector Málaga, edf. Granada, puerta B, 3.° izq. 23740 Andújar. Jaén.

Cambio juegos Spectrum. También cambio y compro libros de código máquina. Enrique Alapont. C/ Maestro Valls 1-19. Valencia (46022). Tl: (96) 367 53 94.

Intercambio programas para el Spectrum 48K. Tengo muchas novedades (Bomb Jack 2, Ace of Aces, Short Circuit y muchos más). Javier Domínguez Tejero. C/ Ramón y Cajal, 22, 1.° D. Santurce. Vizcaya. Tl: (94) 461 80 33.

Vendo ZX Spectrum 48K, interface 1, Microdrive, 2 interfaces para Joystick, 2 joysticks de Spectravideo (Quick Shot I, II), 3 cartuchos de microdrive, 20 cassettes de juego, 10 cintas de cassette especial computador vírgenes. Todo nuevo. Sólo 25.000 pts. Pedro Jesús Arce Guerrero. Urbanización Las Brisas, 1. C/ Eolo. 6. Mairena de Aljarafe. Sevilla. 41927. Tl: 76 20 66.

Intercambio juegos, ideas y mapas del ZX Spectrum. Interesados escribir a: David Álvarez González. C/ Saavedra, 7 6.° E. 33208 Gijón. Asturias.

Club para todos los amigos aficionados al Spectrum y compatibles. Prometemos contestar a todas las cartas. I. Soft. Club. Bda/ Torresoto. C/ Triana, 4. 11401 Jerez de la Frontera. Cádiz. Tl: (956) 32 12 34.

Intercambio programas para el Spectrum Plus con usuarios de toda España, mandar lista a: Agustín Rodríguez Obel. Huelva, 62. Trigueros. Huelva.

Vendo o cambio por cassette o cualquier periférico (menos joystick) todas las revistas INPUT SINCLAIR aparecidas hasta la fecha. En caso de venderlas, por 5.000 ptas. También cambio programas de juegos y utilidades. Juan Gabriel Villena. C/ San Isidro, 27. Padul. Granada. Tl: (958) 79 02 64. Sólo tardes.

Vendo Hisoft Deupac ensamblador-desensamblador Gens-Mons, original y con instrucciones en castellano. Llamar a José Antonio. Lleida. Tl: (973) 76 05 48.

Compro Interface para Spectrum, así como joystick ambos en buen estado y precio asequible. También quisiera contactar con usuarios del Spectrum de alrededores de Navahermosa. L. Esteban Manzanares. C/ Prado, 21. 4515[illegible] Navahermosa. Toledo.

Vendo Spectrum Plus (del año 85) co[illegible] fuente de alimentación, cables, manua[illegible] de instrucciones (español), cinta de de[illegible]mostración (Goldstar en inglés), 3 jue[illegible]gos: The Drive In, Bubble Buster y e[illegible] Underwurlde, con todas las instruccio[illegible]nes para llegar al final sin problema[illegible] (español). Por sólo 20.000 pts. Migue[illegible] García. Tl: (91) 719 24 46. A partir de la[illegible] 15 h.

Vendo simulador de Spectrum y Si[illegible]mon's Basic (en cinta) para CBM-64 co[illegible] manuales incluidos. Sólo 5.000 pts. Pe[illegible]dro J. Gómez Pérez. Avd. de la Raza, 31 3.° A. 21002 Huelva. Tl: (955) 24 20 31[illegible]

Vendo Spectrum 48K, en buen estad[illegible] (con cables, transformador, etc.) cas[illegible]sette marca COMPUTONE, interfac[illegible] Kempston y un joystick, cinta de de[illegible]mostración, programas de juegos, ut[illegible]lidades, aplicaciones, etc. Todo po[illegible] 35.000 pts. A.L. Barros. G. Rubín, [illegible] Pontevedra 36001.

Cambio juegos de Spectrum; bélicos[illegible] de deportes y de aventuras por alguno[illegible] de los siguientes: Terra Cresta, Ja[illegible] Break, Infiltrator o por cargador unive[illegible]sal de código máquina C/M. (Se estu[illegible]diarán ofertas). L. Leonardo Lara Gó[illegible]mez. C/ Rosales Portón, 10. 1.° D. Ceut[illegible]

Vendo ZX Spectrum Plus seminuev[illegible] con todos sus accesorios; cables, trans[illegible]formador y además interface Kepston[illegible] Todos los manuales del ordenador [illegible] dos libros de informática aplicad[illegible] Cinta guía de funcionamiento y tres ci[illegible]tas de juegos. Todo por 26.000 pt[illegible] Adrián Sánchez Gómez. C/ Eustasi[illegible] Amilibia, 4. 3.° B. San Sebastián 2001[illegible] Guipúzcoa. Tl: (943) 46 63 70.

GANADORES DE LOS MEJORES DE INPUT SINCLAIR

En el sorteo correspondiente al número 20 entre quienes escribisteis mandando vuestros votos a los MEJORES DE INPUT han resultado ganadores:

NOMBRE	LOCALIDAD	JUEGO ELEGIDO
Julián Ruiz Herguedas	Valladolid	Gauntlet
Agustín Alarcón Muller	Sevilla	Deep Strike
Adrián Sánchez Gómez	S. Sebastián (Guipúzcoa)	Terra Cresta
Daniel Costa Royo	Barcelona	The Great Escape
Óscar Esteban de Pablo	Villaverde (Madrid)	The Great Escape
Francisco López Urtiaga	Pontevedra	Nuclear Bowls
J. Carlos Adrián García	Portugalete (Vizcaya)	Misterio del Nilo
Carlos Zarzuela Vilafranca	Olesa de Montserrat (Barna)	Livingstone, supongo
José Antonio Piñeiro Vidal	Madrid	The Great Escape
Luis González Fernández	Salamanca	Howard